Un livre de Wikilivres.

LaTeX

Une version à jour et éditable de ce livre est disponible sur Wikilivres,
une bibliothèque de livres pédagogiques, à l'URL :
http://fr.wikibooks.org/wiki/LaTeX

Généralités

LaTeX — prononcer « latèk » ou « latèr », selon les goûts[1] — est un langage de description de document, permettant de créer des documents écrits de grande qualité : principalement livres et articles, mais aussi, courriers, présentations projetées…

Contrairement aux traitements de texte habituels, il n'est pas tel-tel (*WYSIWYG*, en anglais, de *what you see is what you get*) : on ne voit pas à l'écran la manière dont le document sera imprimé ou projeté. Cette mise en forme sera faite par le logiciel : programme appelé `latex` ou `pdflatex`.

Dans un premier temps, l'auteur doit faire confiance au logiciel pour réaliser la mise en page ; celui-ci est configuré pour les canons en la matière, pour appliquer les règles de l'art. Cela donne en général un résultat satisfaisant ; d'ailleurs, l'auteur n'est que rarement typographe, et on peut considérer LaTeX comme un collaborateur spécialisé en typographie, qui prendrait en charge toute la mise en forme du travail, tandis que l'auteur se consacrerait exclusivement au contenu. Chacun sa spécialité !

LaTeX est donc en fait un langage de **préparation de copie** : on donne des *instructions* au typographe virtuel[2].

Le document LaTeX est simplement un fichier de texte pur. Il contient les mots du texte auxquels on ajoute les instructions au typographe virtuel. Les instructions commencent toujours par une barre oblique inverse « \ », également appelée contre-oblique ou barre de fraction inversée (aussi *anti-slash* ou *back-slash*, en anglais). Ces instructions sont aussi appelées des balises, et LaTeX entre dans la catégorie des langages de balisage.

Par exemple, si l'on veut mettre un mot en évidence, on tapera :

```
Mettre un mot en \emph{évidence}.
```

Lorsque le typographe virtuel — le programme `latex` ou `pdflatex` — traitera le fichier, il reconnaîtra la commande `\emph{…}` (de l'anglais *emphasis*, qui signifie *en évidence*) et génèrera :

Mettre un mot en *évidence*.

On remarque au passage qu'en « bonne typographie », la mise en évidence est simplement l'italique.

L'auteur voudra sans doute donner des instructions de plus en plus précises au typographe virtuel, afin que le document produit ressemble à ce qu'il désire ; bref, laisser moins d'initiative au typographe, voire rompre avec les canons. C'est tout à fait possible, mais c'est comme tout : s'il est facile de faire des choses simples, plus on veut faire des choses compliquées, plus il faut connaître d'instructions…

Pourquoi choisir LaTeX ?

On peut se demander pourquoi apprendre un langage d'instructions alors que l'on peut faire la même chose avec la souris en utilisant un traitement de texte. Les raisons qui peuvent amener à choisir LaTeX sont :

- la gratuité ;
- la liberté ;
- le caractère multiplateforme : un même fichier peut être compilé sur un compatible PC sous Microsoft Windows, Linux ou FreeBSD, sur un Apple Macintosh sous macOS, sur une station Sun sous Solaris, … le résultat sera exactement le même ; on peut donc simplement s'échanger ses fichiers, ou bien développer son document sur plusieurs machines différentes ;
- la robustesse : s'il est possible de faire des erreurs en écrivant les instructions ou de ne pas réussir à obtenir exactement ce que l'on veut, en revanche, le programme est très stable et ne plante pas, il n'y a pas de corruption de fichier :
  - la légèreté des fichiers : ce ne sont que des fichiers texte, les images sont des fichiers à côté, on a ainsi des fichiers très légers, peu susceptibles de se corrompre,
  - c'est un langage compilé : contrairement à un logiciel tel-tel, le programme n'a pas à mettre à jour en permanence la pagination, les numéros de page de la table des matières, … ceci est fait lors de la compilation — opération qui génère le document final — ce qui réduit les sources d'erreur ;
- séparation du fond et de la forme : l'auteur peut se consacrer exclusivement au fond, au sens de ce qu'il écrit, et n'est pas distrait par la mise en forme ;
- l'esthétique : le résultat est conforme aux canons de la typographie, en particulier en ce qui concerne les formules mathématiques.

LaTeX, c'est compliqué ?

Certes, il faut apprendre des instructions. Mais on peut se contenter de connaissances minimales :

- le squelette du fichier sera quasiment toujours le même, on peut donc avoir un fichier de base contenant déjà les premières instructions, en particulier l'en-tête ;
- réfléchissez bien : la plupart du temps, vous n'utilisez que très peu d'outils de votre traitement de texte ; de même, vous n'aurez à connaître que peu d'instructions, et dans les cas particuliers, vous pourrez vous reporter à un manuel ;
- le choix d'un éditeur de texte adapté (voir *Installer LaTeX > Choisir un éditeur de texte*) facilite grandement la tâche : il insèrera pour vous les instructions les plus communes à l'aide de la souris (bouton graphique ou menu).

LaTeX ne deviendra compliqué que si vous désirez faire des choses vraiment spéciales, comme par exemple dessiner.

Dans de nombreux cas, les « choses vraiment spéciales » auront été déjà faites par d'autres. Il suffit alors de rechercher sur Internet pour trouver le code, puis simplement de le copier dans votre document. Ces solutions pourront vous être indiquées par des forums (voir *À l'aide ! > Rechercher sur Internet*). Copiez, copiez, copiez sans hésitation, le code est gentiment mis à disposition par son auteur pour ça. Mais il peut ne pas être totalement adapté à votre application ou bien ne pas être « très propre », d'où l'intérêt d'un forum où la multiplicité des contributeurs et l'interaction permettent d'ajuster la solution.

LyX

LyX est un programme permettant d'avoir une interface graphique simple, pseudo-tel-tel, mais avec la stabilité et la beauté de mise en forme de LaTeX. C'est un programme ayant une interface graphique proche de celle des traitements de texte classiques, avec par exemple un bouton *I* et un menu **Format > Italique** pour mettre en italique.

LyX permet d'afficher un aperçu du document, sans avoir toutefois la mise en page exacte (sauts de pages, largeur de page), le résultat est assez proche d'un navigateur Web (la longueur des lignes s'adapte à la largeur de la fenêtre). Le fichier est enregistré dans un format texte propre à LyX, mais on peut aussi lui faire enregistrer des fichiers LaTeX.

LyX utilise LaTeX pour générer le document ; lorsque l'on installe LyX, LaTeX s'installera également (si l'on n'en a pas déjà un). Par contre, comme l'interface graphique est limitée (on ne dispose pas de barres d'outils et de menus configurable « à l'infini »), LyX est limité aux extensions *(packages)* standards. Il n'est pas aussi puissant que le LaTeX programmé « à la main ».

Il devrait toutefois être suffisant à la plupart des applications — mettons 80 % pour respecter la loi de Pareto —, c'est donc un bon moyen de débuter LaTeX. On peut ainsi créer des documents facilement, et inspecter les fichiers LaTeX créés.

Pour le télécharger :

> http://www.lyx.org/

LaTeX et TeX

LaTeX est en fait un produit qui a été développé au milieu des années 1980 à partir d'un produit plus ancien, TeX, développé en 1978.

Il convient de dire que LaTeX est avant tout une encapsulation de certaines fonctions de TeX à destination des utilisateurs. La rédaction d'un document s'en est trouvée grandement simplifiée mais aussi harmonisée. À l'heure actuelle, les besoins principaux sont couverts par le biais d'extensions, également appelées paquetages *(packages)*. La recherche dans les sites officiels peut à ce titre être très utile.

Dans le cas de besoins très particuliers, rien ne vous empêche de programmer vos propres commandes (TeX étant un langage « infini », la seule limitation est la compétence de chacun). La tâche peut être cependant très ardue.

Notes

1. le X est en fait un khi grec, qui se prononce /x/ en fin de mot, comme le *j* espagnol de *jota* ou le *ch* allemand de *Buch* ; mais le terme provient de τέχνη (art, technique), et dans ce mot il se prononce /k/ ; voir Wikipédia : LaTeX > Prononciation
2. rendons à César ce qui lui appartient : la notion de préparation de document est issue de *LaTeX, synthèse de cours et exercices corrigés* de Bitouzé et Charpentier, Pearson Education, 2006, ISBN 2-7440-7187-0 ; et la notion de typographe virtuel est inspirée de la présentation du logiciel LilyPond (http://lilypond.org/web/about/automated-engraving/engraving) [archive].

Voir aussi

Dans Wikibooks

- Programmation TeX

Dans Wikipédia

- LaTeX
- Langage_de_balisage

Premiers pas

Installer LaTeX

Choisir une distribution

La première chose à faire consiste à installer une distribution de LaTeX…

LaTeX est un logiciel libre, vous pouvez donc télécharger une distribution sur Internet, ou bien copier le CD-ROM ou DVD-ROM d'un ami. Certaines versions sont distribuées avec un livre, comme par exemple

- avec *LaTeX par la pratique* (C. Roland, éd. O'Reilly, 1999) : les distribution TeXlive (Windows et Linux), CMacTeX (MacOS 9), emTeX (MS-DOS et OS/2) et DJGPP (MS-DOS et Windows) ;
- avec *LaTeX — Synthèse et cours* (J.-C. Charpentier et D. Bitouzé, éd. Pearson Education, 2006) : la distribution proTeXt pour Windows.

Mais si vous avez ces manuels, vous n'avez sans doute pas besoin de ce wikilivre…

Vous pouvez télécharger les distributions suivantes :

- sous Microsoft Windows :
  - MikTeX : http://www.miktex.org/,
  - http://www.tug.org/protext/, basé sur MikTeX,
  - TeXlive : http://www.tug.org/texlive/,
  - USBTeX (http://www.exomatik.net/LaTeX/USBTeX) [archive] une version portable de LaTeX, moins volumineuse que ProTeXt et plus simple à installer.
- sous MacOS X :
  - MacTeX : http://www.tug.org/mactex/ (recommandé),
  - teTeX (déconseillé : le développement de cette version est arrêté) : http://www.tug.org/tetex/ (peut aussi s'installer *via* `fink`) ;
  - gwTeX de Gerben Wierda : [4] (http://ii2.sourceforge.net/tex-index.html)
  - ozTeX : partagiciel, donc non gratuit[1] : http://www.trevorrow.com/oztex/ ;
- sous les autres Unix (Linux, FreeBSD) :
  - ils sont en général livrés avec une version de LaTeX,
  - teTeX : apt-get install tetex-base sous Debian, voir [5] (http://www.tug.org/tetex/) pour les autres distributions ne possédant pas le paquet dans les dépots.

- distribution multiplateforme
  - TeXlive (http://www.tug.org/texlive/) [archive] : s'obtient avec `apt-get install texlive` sous Debian mais elle est disponible sur le CTAN sous CTAN/systems/texlive (http://mirror.ctan.org/systems/texlive) [archive] et sur le site dédié http://www.tug.org/texlive/.

Le CTAN (http://mirror.ctan.org) [archive] (*Comprehensive TeX Archive Network* c.-à-d. Réseau des Archives complètes de TeX) est un ensemble de sites sur lesquels on trouve la quasi-totalité du matériel lié à TeX. On y trouve, entre autres choses, les principales distributions dans le dossier `systems`.

Notons que si vous installez LyX, celui-ci installe déjà une version de LaTeX si vous n'en avez pas.

Choisir un éditeur de texte

La rédaction en LaTeX consiste donc à créer un fichier de texte. Il vous faut pour cela un éditeur de texte.

Bien sûr, vous pouvez utiliser n'importe quel éditeur de texte générant un fichier au format Unicode. Mais il est intéressant, voire indispensable si l'on veut s'éviter de longues heures de recherche d'erreur, d'avoir un éditeur de texte « orienté LaTeX ».

Outre l'aide à l'édition (on pourra consulter la page *Avec quoi écrire un document HTML ?*, qui détaille bien la problématique), comme la coloration syntaxique qui permet de repérer facilement les instructions et indique les erreurs dans celles-ci, les outils spécialisés permettent d'introduire les instructions les plus courantes avec des clics de souris (boutons de raccourcis, menus), et de lancer directement la compilation du fichier — c'est-à-dire l'analyse et le traitement de la copie par le typographe virtuel, et la génération du résultat au format DVI, PS ou PDF. Cela vous évitera d'ouvrir une fenêtre de commande *(shell)* pour taper les instructions de compilation (ce qui peut être malaisé pour un débutant).

On pourra retenir :

- Emacs ou Vim, qui ne sont pas spécifiques à LaTeX et demandent un apprentissage ;
  - Emacs dispose d'une extension, AucTeX, qui simplifie l'édition de code LaTeX [6] (http://www.parinux.org/ressources/docs/apprendre-latex-grace-a-emacs-auctex#SECTION00022000000000000000) [7] (http://doc.ubuntu-fr.org/emacs) ;
  - Vim dispose d'une extension, LaTeX-suite, qui simplifie l'édition de code LaTeX [8] (http://vim-latex.sourceforge.net);
- Texmaker : http://www.xm1math.net/texmaker/index_fr.html ;
- TeXstudio : http://texstudio.sourceforge.net ;
- TeXnicCenter (uniquement sous Windows) : http://www.toolscenter.org/ ;
- TeXShop : http://www.uoregon.edu/~koch/texshop/ (uniquement sous MacOS X) ;
- TeXworks : http://www.tug.org/texworks/ est multiplateforme et désormais proposé avec la distribution TeXlive;
- Kile : http://kile.sourceforge.net/index.php sous KDE.
- Gummi : http://dev.midnightcoding.org/projects/gummi le code (La)TeX et le rendu PDF automatique, côte à côte (uniquement sous Linux, license libre MIT)

- Winshell, qui prend en charge tous les formats. Correcteur orthographique. http://www.winshell.de/ ;
- WinEdt, qui est un éditeur de texte TeX/LaTeX similaire à Kile. Il fonctionne sous Windows et est proposé sous forme de shareware.

Installer des extensions supplémentaires

Pour faire des choses particulières, vous aurez peut-être besoin d'installer des extensions non standard. Une extension se compose d'un fichier `.sty`, et parfois d'un fichier `.tex` ; les extensions de « l'univers `pstricks` » contiennent parfois des fichiers `.pro`, qui sont des bibliothèques de procédures, fonctions ou variables PostScript. Ces fichiers doivent être mis dans un répertoire qui dépend de l'installation.

Le chemin d'accès au répertoire contenant les extensions est de la forme `[texmf]/tex/latex/`, où `[texmf]` dépend de l'installation. On crée normalement un sous-répertoire par extension (il porte alors le nom de l'extension) ou par thème d'extension. Puis, il faut rafraîchir la liste des extensions afin que LaTeX sache où trouver cette extension.

1. **MikTeX** possède un programme d'installation d'extensions, le *MikTeX Package Manager* (MPM), auquel on peut accéder par le menu **Démarrer** de Microsoft Windows. Si l'on désire installer « à la main », le chemin est en général dans `C:\texmf\tex\latex\`. Si l'on ajoute une extension, il faut en rafraîchir la liste *(refresh)* : menu **Miktex options | General | Refresh now**.

1. De même, **TeXlive** possède `tlmgr` (TeXlive manager) qui permet la mise à jour en ligne et, plus généralement, fournit une interface commune aux différents outils de gestion de la distribution.

1. Pour **MacTeX**, on peut installer les extensions supplémentaires dans :

- `~/Library/texmf` : il s'agit alors d'une extension accessible à l'utilisateur seul ;
- `/usr/local/texlive/texmf-local` : l'extension est accessible à tous les utilisateurs, mais il faut rafraîchir la liste des extensions avec la commande `sudo texhash` (ou `sudo mktexlsr`, qui est un synonyme) depuis l'interpréteur de commandes (`Terminal`). La distribution en elle-même est installée dans `/usr/local/texlive/`*année*`/texmf-dist`, mais ne doit pas être modifiée.

Parfois, l'extension est fournie sous la forme d'un fichier `.ins` et d'un fichier `.dtx`. Il faut alors placer ces fichier dans le répertoire idoine comme indiqué ci-dessus, puis compiler deux fois le fichier `.ins` avec LaTeX[2]. Cela crée alors le fichier `.sty`. Il faut ensuite rafraîchir la liste des extension comme indiqué ci-dessus.

Un certain nombre de distributions sont organisées selon la structure TDS (*TeX Directory Structure* [9] (http://www.gutenberg.eu.org/publications/cahiers/r35-cahiers44-45/187-twg-tds.html)), ce qui permet d'utiliser le gestionnaire d'extensions MikTeX (*MikTeX Package Manager*, MPM) même si l'on n'a pas une distribution MikTeX (par exemple une teTeX ou une TeXLive).

En général, les répertoires de l'installation sont gérés par la bibliothèque `kpathsea`. Pour connaître ces répertoires, il suffit de taper en ligne de commande `texconfig conf` et de regarder les valeurs des variables `TEXMFDIST`, `TEXMFLOCAL` et `TEXMFHOME` ; il ne faut utiliser que les deux dernières pour les installations manuelles.

Annexes

Les différents programmes de compilation

Initialement, le programme de compilation s'appelait `tex`. Les versions de `tex` (les « implémentations ») doivent obéir à un cahier des charges précis.

D'autres programmes ont été développés pour avoir des fonctionnalités légèrement différentes :

- `etex` ;
- `pdftex` ;
- `pdfetex`, qui était un `pdftex` intégrant les possibilités de `etex` ;
- Omega ;
- Alpha ;
- `xetex` : initialement un projet pour la plateforme Apple Macintosh, permettant d'utiliser les fontes OpenType, TrueType, ATT, …
- `luatex` (en cours de développement) : il permet d'utiliser le langage Lua avec du TeX.

Si, pour compiler, on utilise les commandes `latex`, `pdflatex` ou `elatex`, ce sont en fait d'autres programmes qui sont appelés (`tex`, `pdftex`, `etex`) éventuellement avec des options particulières.

Ceci est en général totalement transparent pour l'utilisateur.

Le projet `pdftex` intègre maintenant le code `etex`. C'est le programme par défaut de la plupart des distributions modernes.

Installation avec `apt-get`

La commande `apt-get` concerne la distribution Debian de Linux, ainsi que Fink sous MacOS X.

Fink pour MacOS X peut être téléchargé sur le site Finkproject.org (http://www.finkproject.org/) [archive]. Pour installer teTex, il suffit d'ouvrir une fenêtre d'interpréteur de commandes, et de taper

```
sudo apt-get install tetex
```

Vous pouvez vous reporter à la page des extensions pour le traitement des texte (http://pdb.finkproject.org/pdb/section.php/text) [archive] du projet Fink pour avoir la liste des extensions utiles. Notez que Fink gère lui-même les installations et les dépendances entre les extensions. Si vous voulez ajouter des classes ou des extensions LaTeX, il vaut mieux les ajouter dans un autre répertoire que le répertoire d'installation de teTeX, pour ne pas « casser le *package* ».

Pour désinstaller, il suffit de taper

```
sudo apt-get remove tetex
```

Emacs

Emacs est un éditeur de texte léger et très puissant, mais qui nécessite un apprentissage. Il est de fait déconseillé aux débutants. Il possède un mode spécifique à LaTeX, qui est normalement activé dès lors que l'extention du nom de fichier est `tex`. Il est toutefois possible de l'activer manuellement :

1. appuyez sur la touche Esc, relâchez-la puis appuyez sur la touche X ; la touche Esc est en fait la touche dite « méta », sur certains ordinateurs, on peut utiliser la touche Alt — ou la touche Meta lorsqu'elle existe — et la maintenir appuyée tandis que l'on presse X ;
2. le curseur se retrouve dans la ligne du bas ; tapez `latex-mode` puis appuyez sur la touche Entrée.

Dans la notation conventionnelle Emacs, cette opération est notée `M-x latex-mode RETURN`.

Par défaut, [3] telles que XEmacs ou Aquamacs (voir la documentation spécifique pour chaque application). Pour qu'Emacs fonctionne avec l'encodage latin1, il faut mettre dans le fichier `.emacs.el`[4] :

```
(set-terminal-coding-system 'latin-1)
(set-keyboard-coding-system 'latin-1)
(set-language-environment 'latin-1)
```

Pour plus de détails, on pourra consulter :

- D. Cameron, B. Rosenblatt et E. Raymond, *GNU Emacs*, éd. O'Reilly, 1996, p. 222–227 et 441–442 ;
- D. Cameron, *Emacs précis et concis*, éd. O'Reilly, 1999, p. 33–34.

Syntaxe particulière à certains éditeurs

La syntaxe de LaTeX est indépendante de la plateforme, et donc évidemment de l'éditeur de texte utilisé. Cependant, certains éditeurs de texte admettent des lignes de commentaire particulières, commençant donc par un signe `%`, qui ne sont évidemment pas interprétées par le compilateur mais utilisées par l'éditeur lui-même.

Par exemple, avec TeXShop, le commentaire

```
%!TEX encoding = UTF-8 Unicode
```

indique qu'il faut enregistrer le document au format UTF-8. Avec Emacs, la syntaxe équivalente est [10] (http://www.grappa.univ-lille3.fr /~tommasi/Homepage/EmacsUnicode.html) :

```
% -*- coding: UTF-8; -*-
```

Notes

1. l'utilisateur est invité à payer 30 USD après avoir essayé le produit s'il en est satisfait
2. deux fois parce que si cela peut produire une documentation qui peut contenir des références croisées, voir plus loin
3. name=Emacs ne gère que les caractères en ASCII pur ; cela concerne Emacs « pur » et non pas les adaptations
4. le fichier `.emacs.el` est le fichier de configuration qui se trouve normalement dans le répertoire de l'utilisateur, soit `~/` en Unix ; il est parfois nommé `.emacs` ; s'il n'existe pas, l'utilisateur peut le créer, voir [1] (http://www-verimag.imag.fr/~moy/emacs/dotemacs.html) et [2] (http://www-verimag.imag.fr/~moy/emacs/)

Premier exemple

Nous allons commencer par un exemple de « programme » minimal en LaTeX :

Fichier source

```
\documentclass{article}
\begin{document}
Bonjour, j'\'edite en \LaTeX!
\end{document}
```

Il vous faut ouvrir l'éditeur de texte, recopier (ou utiliser le copier-coller du système d'exploitation) le texte ci-dessus, et l'enregistrer dans un fichier au format « Texte pur » avec un nom portant l'extension `.tex`. On pourra par exemple l'appeler `bonjour.tex`.

Si vous utilisez un éditeur de texte orienté LaTeX, vous devez avoir un bouton ou un menu permettant de générer le fichier de résultat (document final) et de l'afficher. Sinon, il va falloir taper une commande en ligne de commande.

Compilation et visualisation

Avec Texmaker

Vous pouvez appuyer sur la touche F1 du clavier pour lancer la compilation, puis sur la touche F3 pour afficher le résultat.

Vous pouvez aussi utiliser le menu **Tools | LaTeX** pour faire la compilation, et **Tools | View Dvi** pour afficher le résultat.

Avec TeXShop

Vous pouvez utiliser le bouton Typeset (composer), ce qui lance la compilation et affiche le résultat.

Vous pouvez aussi utiliser la combinaison de touches +T, ou encore le menu **Composer | Composer**.

Avec Emacs

Utilisez les touches Ctrl+C puis Ctrl+B (combinaison notée `C-c C-b` dans la notation habituelle Emacs). On ne peut pas visualiser directement le résultat, procéder en ligne de commande. On peut par contre lancer l'impression avec la combinaison Ctrl+C puis Ctrl+P (`C-c C-p`).

En ligne de commande

Si votre éditeur de texte ne gère pas la compilation, il faudra ouvrir un interpréteur de commande *(shell)* :

- sous Windows :
  1. cliquez sur le bouton Démarrer en bas à gauche, et choisissez **Exécuter** ;
  2. dans le champ de texte, taper `cmd` puis appuyez sur Entrée ;
- alternative sous Windows : vous pouvez aussi taper les instructions directement dans le champ de texte de la boîte Exécuter ;
- sous MacOS :
  1. ouvrez le Finder et allez dans le répertoire `Applications > Utilitaires` ;
  2. double-cliquez sur `Terminal.app` (vous pouvez aussi fare glisser l'icône dans le Dock pour l'avoir sous la main).

Dans la fenêtre de l'interpréteur, taper

```
latex bonjour
```

(on peut omettre l'extension `.tex` du nom du fichier). Cela crée un fichier `bonjour.dvi`. Le bon déroulement de la compilation est signalé par le message

```
Output written on bonjour.dvi (1 page, 232 bytes).
Transcript written on bonjour.log.
```

Vous pouvez alors tenter un double-clic sur l'icône du fichier DVI dans votre gestionnaire graphique (Explorateur Windows ou Poste de travail sous Windows, Finder sous MacOS). Sinon, il faut savoir quel programme est disponible pour visualiser les DVI (`yap`, `xdvi`, `dvipreview`, `texshop`…), et taper *`nom-du-programme`* `bonjour` (par exemple `xdvi bonjour`).

En désespoir de cause, vous pouvez utiliser `dvips` et taper

```
dvips bonjour
```

soit cela lancera l'impression, soit cela génèrera un fichier PostScript `bonjour.ps` que vous pourrez lire — normalement — assez facilement. Si

cela lance l'impression, vous pouvez taper

```
dvips bonjour > bonjour.ps
```

ou

```
dvips -o bonjour.ps
```

pour générer le fichier PostScript.

Résultat

Le résultat exact que vous devriez obtenir est :

> Bonjour, j'édite en LaTeX !

Le rendu exact nécessite d'utiliser des images. Pour des raisons pratiques, nous afficherons souvent des rendus approximatifs, sous la forme

Bonjour, j'édite en LaTeX !

Explications

On commence par définir la classe de document, avec l'instruction `\documentclass`. En effet, un livre n'a pas la même mise en page qu'un article de journal. Nous choisissons ici la classe `article`.

Puis, nous indiquons que c'est le début du document avec l'instruction `\begin{document}`.

Nous tapons ensuite le texte, avec les spécificités suivantes :

- pour avoir le « é », nous utilisons `\'e` ; nous verrons comment éviter ceci juste après ;
- pour avoir le logo LaTeX, nous utilisons l'instruction `\LaTeX` ;
- l'espace insécable[1] entre LaTeX et le point d'exclamation est obtenu avec un tilde « ~ ».

L'instruction `\end{document}` marque la fin du document.

Génération d'un fichier PDF

Le format PDF est de plus en plus utilisé pour échanger des fichiers.

Les distributions de LaTeX sont en général livrées avec un programme appelé `pdflatex` qui génère directement un fichier PDF au lieu d'un DVI.

Avec Texmaker

Il suffit de compiler en utilisant la touche F6. La visualisation se fait avec F7. Vous pouvez aussi utiliser le menu **Tools | PDFLaTeX** et **Tools | View PDF**.
Dans certains cas, si vous compilez avec `tex` (et non `pdf(la)tex`), vous pouvez aussi faire générer le fichier PDF en sélectionnant l'option **LaTeX + dvips + View PS** dans la boîte de dialogue des **Préférences | Quick build**.
Si vous avez déjà généré le fichier DVI, vous pouvez le transformer avec la touche F9, ou avec le menu **Tools | DVI → PDF**, et le visualiser comme ci-dessus.
Notez que le touche F9 peut ne pas fonctionner sous macOS X : par défaut, cette touche sert à afficher toutes les fenêtres ouvertes à la fois (« exposé »).

Avec TeXShop

Par défaut, TeXShop utilise `pdftex` ; il génère donc un fichier PDF, et c'est ce fichier qui est affiché.

En ligne de commande

Il suffit de taper

```
pdflatex bonjour
```

Si vous avez déjà généré le fichier DVI, vous pouvez taper

```
dvipdf bonjour
```

La commande `dvipdf` est en fait une macro qui exécute `dvips` pour transformer le fichier DVI en PostScript, puis `ps2pdf` pour transformer le fichier PostScript en PDF. Il existe aussi `dvipdfm`, qui transforme directement le fichier DVI en PDF :

```
dvipdfm bonjour
```

Mentionnons enfin les programmes basés sur Ghostscript (GhostView, GSView ou MacGSView par exemple) qui transforment un PS en PDF, ainsi que sur MacOS X `Aperçu.app` (`Preview.app`).

Remarque

Toutes les solutions ne sont pas équivalentes. En particulier :

- le passage par un fichier PostScript permet d'utiliser certaines extensions qui génèrent directement du code PostScript (comme PSTricks), ce que ne permet pas de faire `pdf(la)tex` ;
- les tailles des fichiers PDF peuvent être différentes ; en général, un fichier obtenu par `pdf(la)tex` est plus gros qu'un fichier obtenue par `dvips` puis `ps2pdf` ;
- mais `pdf(la)tex` est plus simple d'utilisation (une seule opération, mais certains éditeurs de texte génèrent aussi un fichier PDF par `dvips` puis `ps2pdf` en une seule opération), et il permet d'utiliser certaines extensions dédiées (PFG/TikZ) ;
- le format des images que l'on peut intégrer n'est pas le même selon le mode de compilation (PS et EPS pour `dvips` puis `ps2pdf`, PNG et JPEG pour `pdf(la)tex`).

Ces points seront détaillés le moment venu.

Améliorations

Améliorations du jeu de caractères

Vous avez vu que pour faire le e accent aigu « é », nous avons dû taper l'instruction `\'` devant la lettre e. En effet, par défaut, LaTeX ne reconnaît que les caractères latins non accentués des années 1960 (dits « ASCII purs »[2]). Nous pouvons indiquer à LaTeX qu'il doit prendre en compte les caractères accentués de la langue française, afin de pouvoir les saisir directement avec le clavier.

Avant tout, il faut savoir l'encodage de caractères à utiliser. Les systèmes récents ont tendance à travailler avec l'encodage Unicode[3]. Mais des systèmes plus anciens sont plus axées sur des jeux de caractère limités propres à des langues régionales.

Pour pouvoir indiquer l'encodage utilisé, il faut :

- utiliser un éditeur de texte qui permet d'enregistrer autre chose que de l'ASCII pur, ce qui est le cas de tous les éditeurs récents ; on peut en général choisir le codage du fichier lors de l'enregistrement, ou bien dans la boîte de dialogue de configuration de l'éditeur ;
- faire appel à une extension, ou module, ou *package*, appelé `inputenc` *(input encoding*, codage de l'entrée).

Lors de l'appel de l'extension `inputenc`, il faut indiquer un paramètre qui dépend de l'encodage des caractères utilisé pour sauvegarder le fichier. Les trois encodages normalisés les plus courants en europe de l'ouest sont :

- l'encodage Latin-1, ou ISO 8859-1, qui gère les caractères français, notamment ses lettres diacritiquées[4] et ligaturées[5] en minuscules comme en capitales : on utilise le paramètre `latin1`, on écrit donc la commande `\usepackage[latin1]{inputenc}` ;
- l'encodage Latin-9, ou ISO 8859-15, qui gère les caractères français et, cette fois, autant œ que Œ : on utilise le paramètre `latin9`, on écrit donc la commande `\usepackage[latin9]{inputenc}` ;
- l'encodage UTF-8, qui gère l'Unicode (plus d'un million de caractères pour plus de 650 langues) : on utilise le paramètre `utf8`, on écrit donc la commande `\usepackage[utf8]{inputenc}`.

Cependant, tous les caractères Unicode n'ont pas de représentation dans une police donnée ; par exemple, si vous mettez un caractère grec dans votre code source, il n'apparaîtra dans le document final que si la police possède les caractères grecs [11] (http://www.tuteurs.ens.fr/logiciels/latex/langues.html).

Par ailleurs, certains modes ou extensions (mode mathématique et extension `listings` par exemple) gèrent mal les caractères « exotiques ».

Les systèmes d'exploitation ont souvent un codage par défaut ; c'est le format utilisé par l'éditeur fourni avec le système, si l'on ne change aucun paramètre. On peut donc tenter :

- `latin1` pour Linux, FreeBSD, et la plupart des Unix *anciens*;
- `utf-8` pour Linux, FreeBSD, et la plupart des Unix *récents*;
- `cp1252` ou `ansinew` pour Microsoft Windows (les deux sont synonymes) ;
- `applemac` pour un Apple Macintosh & Mac OS X[6].

Notez que cette distinction provient de l'encodage *par défaut* pour les systèmes d'exploitation. Un éditeur de texte donné peut très bien enregistrer le fichier selon un autre encodage, voire, si c'est un éditeur de texte orienté LaTeX, reconnaître l'encodage spécifié et le respecter.

On place donc une instruction `\usepackage[`*`paramètre`*`]{inputenc}` entre les instructions `\documentclass{article}` et `\begin{document}`. Cette zone est appelée zone d'en-tête ou préambule.

Autres améliorations du code source

Par ailleurs, LaTeX applique par défaut les règles de typographie anglaise. Cela ne se voit pas sur cet exemple, mais vous voudrez sans doute utiliser la typographie française. Pour cela, il faudra ajouter, dans le préambule, une instruction pour inclure une extension française. Il en existe plusieurs, nous proposons ici l'extension `babel` avec le paramètre `french` : `\usepackage[french]{babel}`[7]. Il existe d'autres extensions mais qui ne sont pas incluses d'office dans les distributions standard.

On notera que l'on ne place plus de tilde ~ devant le point d'exclamation. Il n'en faut pas si l'on veut que `french` puisse travailler correctement.

On prévoit aussi d'inclure des images (photos, dessins, graphiques). Nous allons donc utiliser l'extension `graphicx`. Nous utiliserons les fontes *extended computer modern* (EC), en ayant recours à l'extension `fontenc` avec l'option T1 (codage des caractères dit « de Cork »), et pour permettre l'affichage correct des caractères diacritiqués français, nous ajoutons les fontes *latin modern* (LM) avec l'extension `lmodern`.

On va aussi indiquer à LaTeX que l'on utilise du papier au format A4 et que l'on utilise une police de corps 11 points[8]. La déclaration de classe devient alors `\documentclass[a4paper, 11pt]{article}`.

Exemple

Ainsi, on aura selon le système d'exploitation, et avec les précautions énoncées ci-dessus (possibilité d'enregistrer avec un encodage différent selon l'éditeur de texte) :

Unix (Linux, FreeBSD, MacOS X)

```
\documentclass[a4paper, 11pt]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{graphicx}
\usepackage[french]{babel}

\begin{document}

Bonjour, j'édite en \LaTeX!

\end{document}
```

MS Windows

```
\documentclass[a4paper, 11pt]{article}

\usepackage[cp1252]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{graphicx}
\usepackage[french]{babel}

\begin{document}

Bonjour, j'édite en \LaTeX!

\end{document}
```

MacOS 8 & 9

```
\documentclass[a4paper, 11pt]{article}

\usepackage[applemac]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{graphicx}
\usepackage[french]{babel}

\begin{document}

Bonjour, j'édite en \LaTeX!

\end{document}
```

Ce squelette peut servir de base pour tous vos fichiers.

Déroulement de la compilation

Si, ce qui est probable, vous utilisez l'outil de compilation de votre éditeur de texte, cette opération sera quasiment transparente. Voici toutefois quelques détails qui pourront vous être utiles.

Le but d'une compilation est de générer un fichier DVI. DVI signifie *device independent*, c'est-à-dire « indépendant du périphérique » (affichage à l'écran, impression, projection). C'est ce fichier DVI qui est affiché par le programme adéquat (en général Yap ou Xdvi), et transformé par la suite en fichier PDF, puisque c'est probablement le format final que vous voudrez.

Durant la compilation, le programme `latex` va analyser le contenu du fichier et éventuellement afficher des messages d'avertissement ou d'erreur. Certains sont bénins, comme « `Underfull box` » ou « `Overfull box` », qui indiquent simplement qu'il y a trop ou pas assez de texte dans une zone pour respecter les canons de la mise en page.

Certaines erreurs correspondent à des fautes dans l'orthographe des instructions — d'où l'utilité d'utiliser, lorsque c'est possible, les outils fournis par l'éditeur de texte — ou dans la syntaxe, par exemple l'oubli de `\end{document}` en fin de document. Certaines de ces erreurs vont donner un résultat étrange, non conforme à ce que vous désirez. Mais d'autres empêcheront la compilation de se terminer, vous n'aurez alors pas de résultat.

Ces erreurs et avertissements sont affichés à l'écran, et sont mis dans un fichier portant le même nom que votre fichier source, mais avec l'extension `.log`.

Par ailleurs, `latex` foliote (numérote) automatiquement les pages, les figures, tableaux, appelés de manière générale « objets », et il est possible de faire des références croisées (de type « cf. figure 18 p. 50 »). Il peut donc être nécessaire de faire deux compilations successives :

- lors de la première compilation, il va numéroter les pages et trouver les pages correspondant aux objets pointés, qu'il place dans un fichier auxiliaire ; il ne pourra par contre pas créer les références, puisqu'il ne sait pas encore où se trouvent les objets ni quels sont leurs numéros, il va donc mettre des erreurs « `Reference undefined` » ;
- lors de la deuxième compilation, il lit le fichier auxiliaire et place les références.

Le fichier auxiliaire porte le même nom que votre fichier source, mais avec l'extension `.aux`.

Certains éditeurs de texte font exécuter automatiquement les deux compilations lorsque c'est nécessaire.

Voir aussi À l'aide !.

Notes

1. L'utilisation de l'espace insécable permet de garder le point d'exclamation sur la même ligne que le mot qui précède, si jamais le mot se trouvait en fin de ligne. On utilise une espace insécable avant les ponctuations doubles (;:?!), à l'intérieur des guillemets français **« »**, pour garder un nombre à côté de son unité…
2. Pour se faire une idée des limites de l'ASCII on peut lire le wikilivre Les ASCII de 0 à 127
3. Pour se faire une idée des possibilités offertes par l'Unicode on peut lire le wikilivre À la découverte d'Unicode
4. avec accent, cédilles, …
5. æ mais pas œ ; on pourra donc saisir « `ex-æquo` » mais il faudra « `\oe uf` » pour obtenir « œuf »
6. jusqu'à MacOS 9, le codage applemac était le codage par défaut ; avec MacOS X, le codage par défaut de TextEdit est utf8, et il peut être changé dans le menu **TextEdit | Préférences | Ouverture et enregistrement** (on peut aussi choisir un encodage particulier au moment de l'enregistrement) ; chaque éditeur de texte gère son encodage
7. On peut encore utiliser les paramètres `frenchb` et `francais` qui ont exactement le même effet. On les trouve dans les exemples de sources.
8. c'est-à-dire que la hauteur entre le bas d'un jambage, comme la lettre « p », et le haut d'une hampe, comme la lettre « t », vaut 11 points anglo-saxons soit environ 3,9 mm, cf. Wikipédia : *Point (unité)*

Lettre

La classe `lettre` (variante de la classe `letter`) est un excellent exemple pour illustrer la séparation du fond et de la forme. Considérons le fichier suivant :

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

\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{eurosym}
\usepackage[frenchb]{babel}
\usepackage{numprint}

\begin{document}

\begin{letter}{Service des réclamations}
\name{Jean Râleur}
\address{Jean Râleur\\4, rue du Bac à sable\\80886 Sassone-le-Creux}
\lieu{Sassone-le-Creux}
\telephone{01 02 03 04 05}
\nofax

\def\concname{Objet :~} % On définit ici la commande 'objet'
\conc{rétractation}
\opening{Madame, Monsieur,}

Vous m'avez démarché la semaine dernière
pour me proposer l'édition de luxe de l'encyclopédie Wikipédia,
pour la somme de \numprint{5000}~\euro.
Conformément à la loi m'accordant un délai de rétractation de 7~jours,
je renonce à mon achat et demande le remboursement de la somme versée.


\closing{Je vous prie d'agréer,
Madame, Monsieur,
mes salutations distinguées.}

\end{letter}

\end{document}
```

La compilation donne :

Jean Râleur
4, rue du Bac à sable
80886 Sassone-le-Creux
Tél. 01 02 03 04 05

Sassone-le-Creux, le 5 mai 2008

Service des réclamations

Objet : rétractation

Madame, Monsieur,

Vous m'avez démarché la semaine dernière pour me proposer l'édition de luxe de l'encyclopédie Wikipédia, pour la somme de 5 000 €. Conformément à la loi m'accordant un délai de rétractation de 7 jours, je renonce à mon achat et demande le remboursement de la somme versée.

Je vous prie d'agréer, Madame, Monsieur, mes salutations distinguées.

Jean Râleur

On voit que la mise en forme est entièrement faite par LaTeX, et dans le cas présent par la classe `lettre` : la plupart des commandes et environnement utilisés ici sont spécifiques à cette classe.

La documentation détaillée pour la rédaction de courriers avec la classe `lettre` est sous la forme d'un fichier PDF appelé `letdoc.pdf` [12] (http://mirrors.ctan.org/macros/latex/contrib/lettre/lettre.pdf).

Éléments de base

Bloc

Dans LaTeX, un bloc est une portion de texte comprise entre deux accolades « {…} ». Souvent, une instruction agit sur un bloc, par exemple :

```
\emph{texte}
```

ou bien un bloc sert à limiter la zone sur laquelle agit une instruction de portée « infinie », par exemple

```
{\em texte}
```

On utilisera parfois le bloc vide

```
{}
```

> Les accolades n'apparaissent pas sur le rendu final

(voir la section suivante pour mettre des accolades dans le texte final).

Caractères réservés

Dans le premier exemple, nous avons vu que la contre-oblique « \ » servait à indiquer les instructions, les accolades « {} » un bloc et que le tilde « ~ » indiquait l'espace insécable. Si nous voulons faire figurer ces caractères dans le document final, on ne peut donc pas les taper tels quels dans le fichier source. Il existe d'autres caractères réservés, en voici la liste :

```
{ } % # $ ^ ~ & _ \
```

l'avant-dernier caractère est le tiret de soulignement ou *underscore* (touche 8 sur un clavier de PC, ou touche à droite de la parenthèse fermante sur un clavier Mac).

Si nous voulons utiliser ces caractères, il faudra entrer :

- `\{` pour {
- `\}` pour }
- `\%` pour %
- `\#` pour #
- `\$` pour $
- `\textasciicircum` pour ^ (mnémotechnique anglais : *text, ASCII, circumflex*) ;
- `\textasciitilde` pour ~ (mnémotechnique anglais : *text, ASCII, tilde*) ;
- `\&` pour &
- `\_` pour _
- `\textbackslash` pour \ (mnémotechnique anglais : *text, backslash*).

Les commandes avec un caractère réservé peuvent être suivies par n'importe quoi, il n'y a pas d'ambiguïté. Par contre, les commandes constituées de lettres ne peuvent pas être suivies de lettres : comment savoir où s'arrête la commande ? Elles peuvent par contre être suivies d'un signe autre qu'une lettre : un nombre, un signe de ponctuation…

Si l'on veut, dans le document final, accoler une lettre au caractère obtenu avec une telle commande, il faut indiquer la fin de la commande :

- soit laisser une espace après la commande ; cette espace sera ignorée pour le rendu final ;
- soit placer une paire d'accolades vides {}.

Par exemple

- `\textasciicircumA` génère une erreur ;
- `\textasciicircum␣A` et `\textasciicircum{}A` donnent « ^A » ;
- `\textasciicircum{}␣A` donne « ^ A ».

Le caractère « ␣ » sert à indiquer l'espace, pour clarifier les choses. On note ici que le fait que le fichier contienne un caractère espace (obtenu avec la barre d'espacement du clavier) ne signifie pas que le document final aura une espace[1].

Diacritiques et ligatures

Au chapitre *premier exemple*, nous avons vu qu'avec l'extension `inputenc`, nous pouvions choisir le jeu de caractères et donc utiliser les caractères accentués, les caractères de la langue française et de manière plus spécifique diacritiqués, du clavier. Le problème se pose lorsque le clavier ne

permet pas de faire les caractères que nous voulons, par exemple :

- les capitales accentuées[2] comme « É » ;
- les lettres des langues étrangères au clavier : accents français avec un clavier anglais, lettres danoises avec un clavier français, …

Voici quelques exemples :

- `\'e` pour é ;
- `\`e` pour è ;
- `\^\i` pour î ;
- `\`A` pour À ;
- `\"e` pour ë ;
- `\c{c}` ou `\c c` pour ç,

et ainsi de suite. On remarque que :

- le diacritique est une commande qui est suivie directement par la lettre :
  - `\'` (contre-oblique et apostrophe) pour l'accent aigu,
  - `\`` (contre-oblique et apostrophe inversée) pour l'accent grave,
  - `\^` (contre-oblique et chapeau) pour l'accent circonflexe,
  - `\"` (contre-oblique et guillemet double anglais) pour le tréma,
  - `\c` (contre-oblique et c) pour une cédille ; la lettre cédillée doit être mise entre accolades, ou bien séparée d'une espace ;
- pour mettre un accent sur le i minuscule, il faut utiliser `\i` (contre-oblique, i), afin de supprimer le point sur le i avant de placer l'accent.

On peut aussi utiliser les ligatures :

- `\oe` pour œ ;
- `\OE` pour Œ ;
- `\ae` pour æ ;
- `\AE` pour Æ.

Les physiciens pourront avoir recours à `\AA` pour Å.

Le problème des commandes composées de lettres (comme `\c`, `\i`, `\oe` et `\AA`) est le même que ci-dessus. Par exemple, il faut taper

- `c\oe ur` ou `c\oe{}ur` pour obtenir « cœur » ;
- `fa\^\i t` ou `fa\^\i{}t` pour obtenir « faît » ;
- `\c ca` ou `\c{c}a` pour obtenir « ça ».

Le dernier exemple est un peu différent : `\c{c}` indique que la commande `\c` s'applique à `{c}`.

Cette précaution est par contre inutile si le caractère suivant n'est pas une lettre. Comment alors indiquer que l'on veut une espace après ? Deux solutions :

- soit on indique explicitement la présence d'une espace dite « justifiante » (c'est-à-dire non insécable et de dimension variable pour s'ajuster à la longueur de la ligne et au nombre de caractères), avec la commande `\ ` (contre-oblique, espace) ;
- soit on indique la fin de la commande avec des accolades vides `{}`, le tout suivi d'une espace.

Par exemple :

```
On peut obtenir \OE\ avec \textbackslash OE, et \AE{} avec \textbackslash{}AE.
Le symbole de l'angstr\"om est \AA.
```

donne

On peut obtenir Œ avec \OE, et Æ avec \AE. Le symbole de l'angström est Å.

Quelques autres caractères

Voici quelques autres caractères que l'on peut obtenir.

Caractères LaTeX

Saisie	Caractère	Note
`\copyright`	©	
`\dag`	†	obèle
`\ddag`	‡	
`\dots`	…	points de suspension (plus espacés que trois points)
`\o`	ø	

\O	Ø	
\P	¶	
\pounds	£	livre sterling
\S	§	paragraphe
\ss	ß	eszett (lettre allemande)
\textbar	|	tube
\textperiodcentered	·	point centré
\textregistered	®	*registred*
\texttrademark	™	*trade mark*
\textvisiblespace	␣	

Certains caractères ont une saisie simplifiée.

Caractères LaTeX

Saisie	Caractère	Note
?`	¿	alternative : \textquestiondown
!`	¡	alternative : \textexclamdown
-	-	division (trait d'union)
--	–	tiret demi-cadratin, alternative : \endash
---	—	tiret cadratin, alternative : \emdash
<<	«	guillemet français ouvrant, avec le codage de polices T1 mais voir ci-dessous
>>	»	guillemet français fermant, avec le codage de polices T1 mais voir ci-dessous

Les symboles ci-dessus peuvent aussi être mis directement dans le fichier `.tex` si vous utilisez un codage adéquat, comme l'ISO Latin 1 ou l'UTF8 (cf. *Premier exemple > Améliorations du code source*). Notez que l'utilisation des guillemets français tels quels ne provoque pas un affichage correct des espaces en français, dans le cas d'une citation, il faut donc utiliser `\og` et `\fg` à la place (voir ci-après).

L'euro (la monnaie) a été introduite récemment par rapport à l'histoire de LaTeX. Elle n'est donc pas disponible dans les fontes de base. Pour l'obtenir, il faut :

- charger l'extension `eurosym`, en mettant `\usepackage{eurosym}` dans le préambule ;
- dans le texte, utiliser la commande `\euro` (éventuellement `\euro{}`).

Cas du point d'abréviation anglais

En typographie anglaise, l'espace après un point final de phrase est plus grande que l'espace après un point d'abréviation. Par défaut, LaTeX considère qu'un point est un point final, mais il peut reconnaître dans certains cas les points d'abréviation (comme par exemple dans `U.F.O.`). Si l'on veut forcer un point d'abréviation, il faut utiliser `.\@` (point, contre-oblique, arobase).

Exemple

```
U.F.O.\@ means ``unidentfied flying object''.
```

Dans certains cas, il vaut mieux utiliser le tilde, espace insécable, qui permet d'avoir le bon espacement *et* de ne pas avoir de coupure en fin de ligne :

Exemple

```
And here's to you Mrs.~Robinson
```

À l'inverse, si l'on veut forcer un point de fin de ligne (par exemple phrase se terminant par une abréviation, le dernier point d'abréviation étant aussi point de fin de phrase), on peut écrire `\@.`, ou bien mettre le mot entre accolade.

Exemple

```
This is a U.F.O\@.
```

ou bien

```
This is a {U.F.O.}
```

Tout ceci ne s'applique pas en typographie française (l'espace est identique quel que soit le type de point), et l'extension `babel` ajuste automatiquement la taille de l'espace après un point (voir plus loin).

Extension française

L'extension française utilisée fournit des commandes supplémentaires, facilitant l'édition.

Voici quelques exemples :

Commandes communes à `babel` option `french` et à `frenchle`

- `\og` : « suivi d'une espace insécable ;
- `\fg` : » précédé d'une espace insécable ;

Commandes de `babel` option `french`

- `\degres` : ° ;
- `\ier` : er ;
- `\iere` : re ;
- `\iers` : ers ;
- `\ieres` : res ;
- `\ieme` : e ;
- `\iemes` : es ;
- `\bsc{…}` : petites capitales, le mot n'étant pas césuré.

Espaces et changements de ligne

On remarque dans l'exemple ci-dessus que le fait que l'on ait changé de ligne dans le fichier source ne provoque pas de changement de ligne dans le résultat final. Le changement de ligne simple équivaut à une espace. Par contre, si on laisse une ou plusieurs lignes vides, cela indique un changement de paragraphe, caractérisé par un retour à la ligne et :

- soit un alinéa rentrant (indentation), qui est le cas général en typographie française et anglaise ;
- soit par un interligne plus grand, solution souvent retenue pour des ouvrage techniques et popularisée par les navigateurs Web et le traitement de texte Microsoft Word, pour lesquels c'est la mise en page par défaut.

On peut laisser plusieurs lignes vides, cela équivaut à une seule ligne vide.

De même, si l'on met plusieurs espace, c'est comme s'il n'y en avait qu'une seule.

Concernant les blancs entre les mots (espaces, au féminin), il existe plusieurs types d'espaces :

- espace justifiante (de dimension variable) : obtenue par une ou plusieurs pressions sur la barre d'espacement et/ou une pression sur la touche Entrée, ou bien par `\␣` ;
- espace insécable : ~ (tilde) ;
- petite espace : `\,` ;
- espace fine : `\/` : espace encore plus petit, permettant de régler des problèmes de débordement de l'italique (`\emph{…\/}`), ou bien d'empêcher les ligatures.

Pour introduire une espace horizontale d'une longueur donnée, on utilise la commande `\hspace{`*`longueur`*`}` où *longueur* est un nombre avec une unité accolée :

- `mm` (millimètre) ;
- `cm` (centimètre) ;
- `pt` (point anglo-saxon) ;
- `dd` (point Didot) ;
- `ex` (hauteur d'x) ;
- `em` (cadratin).

Les unités `ex` et `em` sont proportionnelles au corps de la police :

- la hauteur d'x, parfois appelée à tort « hauteur d'œil », est la hauteur d'un bas de casse (lettre minuscule) sans hampe ni jambage, comme le *x* ;
- le cadratin est égal au corps de la police.

Par exemple :

> `\hspace{1cm}` pour une espace de un centimètre ;
> `\hspace{0,5em}` pour une espace d'un demi cadratin.

On peut utiliser un point ou une virgule comme séparateur décimal.

La commande `\hfill` introduit un espace « ressort » : il pousse ce qu'il y a à gauche et à droite pour occuper tout l'espace restant sur la ligne. S'il y a plusieurs commandes `\hfill` sur la même ligne, les espaces sont de même largeur.

De même, si l'on veut espacer des paragraphes, on utilise la commande `\vspace{`*`longueur`*`}`. On peut également utiliser les commmandes :

- `\medskip` pour sauter une ligne « normale » ;
- `\smallskip` pour un « petit » saut de ligne ;
- `\bigskip` pour un « grand » saut de ligne.

La commande `\vfill` introduit un espace « ressort » : il pousse ce qu'il y a au dessus et en dessous pour occuper tout l'espace restant sur la page. S'il y a plusieurs commandes `\vfill` sur la même page, les espaces sont de même largeur.

Structuration du code source

Comme pour tout code source de programme, on a intérêt à structurer le fichier LaTeX afin de retrouver facilement les passages que l'on veut.

LaTeX n'étant pas wysiwyg, on peut organiser le texte comme l'on veut, ce qui facilite sa lecture à l'écran ; en particulier, on peut introduire un retour de ligne entre chaque portion de phrase, entre chaque proposition. Le fait de pouvoir sauter des lignes entre les paragraphes et mettre des espaces ou tabulations à profusion en début de ligne permet de donner une « forme » au code qui rend son analyse facile (notion d'indentation).

Par ailleurs, on peut introduire des commentaires en utilisant le signe `%`.

Notons que les commentaires permettent aussi :

- d'effacer une portion de texte sans la perdre : on met les lignes en commentaires (on place un « % » à leur début) pour enlever le texte du document final, et il suffit de les décommenter (d'enlever le « % ») pour faire réapparaître le texte dans le document final ;
- de faire comme si l'on ne revenait pas à la ligne, en mettant un « % » à la fin de la ligne.

Notes

1. en typographie le mot espace est au féminin lorsqu'il désigne un écart horizontal
2. en bonne typographie, on met les accents sur les capitales. Notez que certains systèmes (comme GNU/Linux) gèrent nativement les majuscules accentuées. Pour cela, activez le verrouillage majuscule (touche au-dessus du Shift) et appuyez sur la touche accentuée concernée.

Voir aussi

Liens externes

- The Comprehensive LaTeX Symbol List (http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf) [archive] (fichier PDF, 105 p, 3 Mo)

Les classes

La première chose que vous devez faire quand vous éditez un document LaTeX, c'est de lui préciser sa classe, c'est-à-dire le type de document que vous souhaitez réaliser. Il existe plusieurs classes, qui sont spécifiées par la commande `\documentclass[options]{classe}`

Les différentes classes disponibles par défaut sont :

- `article` : pour des articles destinés à la publication et ne contenant que quelques pages ;
- `report` : pour des documents un peu plus longs contenant plusieurs chapitres, comme des mémoires de thèse ;
- `book` : pour de véritables livres, de plusieurs centaines de pages.
- `slides` : pour faire des présentations sur transparents.

Il existe aussi d'autres classes moins répandues :

- `beamer` : pour faire des présentations utilisant la magnifique extension `beamer` ;
- `lettre` : pour faire des lettres au format français (classe écrite par l'Observatoire de Genève[1]).
- `memoir` : pour écrire des mémoires, par exemple de fin d'étude.

Le choix de la classe va déterminer un certain nombre de paramètres par défaut, comme par exemple les marges, mais aussi fournir des instructions supplémentaires spécifiques.

Classes book et report

La classe `book` est prévue pour faire un livre. Les spécificités du livre sont les suivantes :

- il dispose d'une page de titre séparée, suivie d'une page blanche ;
- il peut se décomposer en parties, chapitres, sections, sous-sections, sous-sous-sections, paragraphes et sous-paragraphes ;
- les parties et chapitres commencent sur une page impaire (« belle page ») ;
- les marges sont assez grandes pour permettre une lecture aisée (par rapport à la quantité de texte).

La classe `report` est similaire à la classe `book`, mais les chapitres ne commencent pas nécessairement en belle page, et l'on ne peut pas utiliser certaines fonctions comme `\frontmatter`, `\mainmatter` et `\backmatter`. Par contre, elle dispose d'un environnement `abstract` permettant la mise en forme automatique d'un résumé.

La structure typique d'un livre sera :

```
\documentclass[a4paper, 11pt]{book}
\usepackage[''paramètre'']{inputenc}
\usepackage{graphicx}
\usepackage[frenchb]{babel}
\title{''Titre du livre''}
\author{''Nom de l'auteur''}
\date{''date de fin de rédaction''}

\begin{document}
\maketitle

 ''Corps du livre''

\tableofcontents

\end{document}
```

Nous avons ici trois instructions donnant des informations sur le livre :

- `\title` pour le titre ;
- `\author` pour le nom de l'auteur ;
- `\date` pour la date.

l'instruction `\maketitle` crée la page de titre à partir des informations données par les instructions ci-dessus.

L'instruction `\tableofcontents` crée automatiquement la table des matières (ce qui nécessite une double compilation), à partir des parties, chapitres, sections, … définis dans le corps du livre. En typographie française, la table des matières est à la fin.

Classe article

Par rapport à un livre, un article :

- a son titre sur la même page que le début du texte ;
- a des marges plus étroites (mise en page dans un journal) ;
- a moins de subdivisions de texte : il n'y a pas de chapitre.

La structure de base d'un article est similaire à celle d'un livre (on ne peut cependant pas créer de chapitre) ; seul le résultat diffère :

```
\documentclass[a4paper, 11pt]{article}

\usepackage[''paramètre'']{inputenc}
\usepackage{graphicx}
\usepackage[frenchb]{babel}

\begin{document}

\title{''Titre de l'article''}
\author{''Nom de l'auteur''}
\date{''date de fin de rédaction''}

\maketitle

\tableofcontents

''Corps de l'article''

\end{document}
```

Classes spécifiques

Certaines publications scientifiques acceptent les soumissions au format LaTeX ; c'est parfois même le format recommandé. Ils fournissent en général une classe spécifique, avec la documentation requise pour l'utiliser.

Informations sur l'œuvre

Titre

Le titre est nécessairement composé d'un seul alinéa (un seul paragraphe). Si l'on veut introduire des retours de ligne, il faut utiliser `\\`. Si l'on veut « sauter des lignes », il faut utiliser la commande `\vspace` ou une commande similaire (cf. *Éléments de base* > *Espaces et changements de ligne*).

On peut y inclure des images.

Auteur

Si l'ouvrage comporte plusieurs auteurs, ceux-ci sont tous mis dans le bloc de l'instruction `\author`, mais sont séparés par l'instruction `\and`.

Le nom d'un auteur peut être suivi d'une instruction `\thanks{…}`, qui permet de lui associer une note de bas de page.

Par exemple :

```
\author{
   C. \bsc{Dang Ngoc Chan}
   \and
   C. \bsc{Huvier}
   \and
   J.-F. \bsc{Dinhut}\thanks{à qui l'on adressera la correspondance}
}
```

Date

Si l'on n'utilise pas l'instruction date, LaTeX met la date de la compilation. Pour ne pas avoir de date, il faut définir une date vide :

```
\date{}
```

Table des matières - Généralités - Premiers pas - Structure du document - Gestion de la bibliographie - Tableaux - Images - Éléments flottants et figures - Mise en forme du texte - Choix de la police - Mise en page - Mathématiques - Gestion des gros documents - Faire des présentations - Arts et loisirs - Dessiner avec LaTeX - Créer une extension ou une classe - Programmer avec LaTeX - *Annexes* - Vade mecum - Conversion - Glossaire de typographie - Index - Commandes - Liens externes

Notes

1. la documentation est dans un fichier appelé `letdoc.ps`

Structuration du texte

Parties d'un texte

Un texte est typiquement découpé en parties, chapitres et sections. Cela n'est pas nécessaire pour un texte court, mais même un court article possède en général deux ou trois sections. Pour un livre, on aura au moins deux chapitres.

Pour structurer un texte, on utilise des instructions qui définissent le titre et le « niveau de titre ». La hiérarchie est :

1. partie : l'instruction est `\part{`*`Titre de la partie`*`}` ;
2. chapitre : `\chapter{`*`Titre du chapitre`*`}` ; **n'existe pas avec la classe `article`** ;
3. section : `\section{`*`Titre de la section`*`}` ;
4. sous-section : `\subsection{`*`Titre de la sous-section`*`}` ;
5. sous-sous-section : `\subsubsection{`*`Titre de la sous-sous-section`*`}` ;
6. paragraphe : `\paragraph{`*`Titre du paragraphe`*`}` ;
7. sous-paragraphe : `\subparagraph{`*`Titre du sous-paragraphe`*`}`.

Il existe des versions « étoilées » de ces commandes, qui ne génèrent pas de numéro de partie/chapitre/section : `\part*`, `\chapter*`, … Par exemple, la préface pourra être introduite par

```
\chapter*{Préface}
```

Voici à titre d'exemple ce que pourrait être le début de Michel Strogoff de Jules Verne :

```
\documentclass[a4paper, 11pt]{book}

\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[frenchb]{babel}

\begin{document}

\title{Michel Strogoff}
\author{Jules \bsc{Verne}}
\date{\oldstylenums{1875}}

\maketitle


% ******************
% *                *
% * Premiere partie *
% *                *
% ******************

\part*{Première partie}

% ********************
% * Chapitre premier *
% ********************

\chapter{Une fête au palais-neuf.}

\og Sire, une nouvelle dépêche.
[…]

\tableofcontents

\end{document}
```

On notera ici qu'en typographie française, la table des matières est à la fin de l'ouvrage.

La génération de la table des matières nécessite deux compilations successives.

Dans le cas d'un livre (classe `book`), on dispose de trois commandes de structuration supplémentaire :

- `\frontmatter`, qui indique tout ce qui se trouve avant la matière principale (page de titre, dédicace, avant-propos, introduction, préface) ; les parties ne sont pas numérotées mais figurent dans la table des matières, et les pages sont numérotées en chiffres romain bas-de-casse ;
- `\mainmatter`, qui introduit le corps de l'ouvrage ;
- `\backmatter` qui introduit les index et tables des matières.

```
\documentclass[a4paper, 11pt]{book}

[...]

\begin{document}
```

```
\frontmatter

\title{Ma vie, mon œuvre}
\author{Moi-même}}
\date{\today}

\maketitle

\mainmatter

\chapter{Avant ma naissance}

[...]

\backmatter

\tableofcontents

\end{document}
```

Notons que les classes `article` et `report` disposent d'un environnement `abstract` permettant la mise en forme d'un résumé :

```
\begin{abstract}
   Résumé du document
\end{abstract}
```

Changement de page

Le changement de page est géré automatiquement par LaTeX. Mais vous pouvez forcer un saut de page avec l'instruction `\newpage`.

On peut aussi se contenter de donner une indication à LaTeX :

- `\pagebreak` encourage un changement de page à cet endroit-là ;
- `\nopagebreak` décourage un changement de page.

LaTeX ne suivra pas systématiquement ces recommandations, en fonction des blancs que cela laisse en bas de page ou des débordements en fin de page.

Fonction d'une portion de texte

Une portion de texte peut avoir une fonction particulière : on utilise alors une instruction pour indiquer au typographe virtuel de mettre en forme cette portion en fonction de sa fonction. C'est le cas par exemple des titre évoqués ci-dessous.

Si une portion de texte doit être mise en emphase, on utilisera l'instruction `\emph{`*`texte en emphase`*`}`. On évitera la syntaxe `{\em texte en emphase}` que l'on pourra trouver dans de vieux documents (voir la section *Mise en forme du texte*).

Si une portion de texte est une liste non numérotée, on l'encadrera par `\begin{itemize}`…`\end{itemize}`, et chaque élément énuméré — chaque article, anglais *item* — sera introduit par l'instruction `\item`, par exemple :

```
\begin{itemize}
   \item premier élément;
   \item deuxième élément;
   \item troisième élément.
\end{itemize}
```

(voir la section *Listes structurées*).

Nous remarquons ici que comme le nombre d'espaces n'a pas d'importance, nous avons utilisé des espaces multiples pour faciliter la lecture du code. Par ailleurs, nous n'avons à aucun moment indiqué de quelle manière le texte allait être mis en forme (en l'occurrence : composition en sommaire, avec un tiret en début d'alinéa).

Il faut mettre ici l'accent sur la notion de séparation du fond et de la forme. On ne devrait jamais indiquer directement la mise en forme, mais indiquer simplement la fonction. Le problème est que, par défaut, LaTeX ne possède qu'un nombre limité de fonctions de texte, qui ne couvrent pas nécessairement vos besoins. Il va donc falloir créer vos propres instructions, ou commandes personnelles, et pour cela connaître les instructions de mise en forme de LaTeX. Ainsi, par exemple, si vous avez à écrire des noms de programmes, vous créerez une commande `\nomprog`, dans laquelle vous indiquerez d'utiliser une police à chasse fixe, ce qui donnera :

Code source

```
Le programme \nomprog{yap} sert à visualiser des fichiers DVI.
```

Rendu

Le programme `yap` sert à visualiser des fichiers DVI.

On pourrait utiliser directement les fonctions de mise en forme de LaTeX dans le corps du document, mais cela amènerait des confusions.

Listes structurées

LaTeX compose les listes en sommaire, c'est-à-dire que le texte est justifié mais que la première ligne est saillante.

Listes numérotées

L'environnement `enumerate` permet de créer des listes numérotées. Chaque élément d'une liste numérotée est introduit par la commande `\item` :

```
\begin{enumerate}
   \item Premier élément;
   \item Deuxième élément.
\end{enumerate}
```

La numérotation est automatique.

> **Remarque :** en typographie française, on met une capitale au début de chaque alinéa commençant par un numéro ou une lettre de classification suivie d'un point, ce qui est le cas des listes numérotées.

Listes non numérotées

Les listes non numérotées sont fournies par l'environnement `itemize`. Elles fonctionnent comme les listes numérotées :

```
\begin{itemize}
   \item premier élément;
   \item deuxième élément.
\end{itemize}
```

En typographie française, les lignes commencent par un tiret long (ou « sur quadratin », à ne pas confondre avec le trait d'union ou le signe moins) « — » ; en typographie anglaise, elles commencent par un disque plein (ou « point ») « • » ou par un tiret moyen (ou « sur demi-quadratin ») « – ». Le caractère initial est géré directement par LaTeX en fonction de la langue déclarée lors de l'appel de l'extension `babel`, et du niveau de liste (liste dans une liste par exemple).

> **Remarque :** en typographie française, on met une minuscule en début d'alinéa tireté (qui commence par un tiret), ce qui est le cas des listes non numérotées.

Descriptions

L'environnement `description` permet d'associer une définition à un terme :

```
\begin{description}
   \item[terme1] définition1;
   \item[terme2] définition2.
\end{description}
```

Imbrication

On peut imbriquer les listes.

```
\begin{enumerate}
   \item Voici le premier élément;
   \item le deuxième élément se décompose en
       \begin{itemize}
          \item élément a,
          \item élément b,
          \item élément c;
       \end{itemize}
   \item le troisième élément clôt tout ceci.
\end{enumerate}
```

1. Voici le premier élément ;
2. le deuxième élément se décompose en
   1. élément a,
   2. élément b,

3. élément c ;
3. le troisième élément clôt tout ceci.

Notes de bas de page

Pour introduire une note de bas de page, on utilise l'instruction `\footnote`. Elle se place après le mot où l'on veut mettre l'appel de note, et le contenu de la note se trouve dans un bloc accolé à l'instruction, par exemple :

```
La SNCF\footnote{Société nationale des chemins de fer français} fournit
un service de transport de voyageurs et de fret.
```

Donne

La SNCF[1] fournit un service de transport de voyageurs et de fret.

1. Société nationale des chemins de fer français

La numérotation se fait de manière automatique. Notez qu'en typographie française, dans un tableau, la note se situe en bas du tableau avec une numérotation spécifique (par exemple des lettres) et non pas au bas de la page. Le nombre de notes étant en général réduit, on gèrera cela « à la main » (mais il existe des solutions permettant de gérer les notes de tableaux automatiquement).

Références

Lorsque l'on fait référence à une partie du texte, il peut être utile d'indiquer le numéro de chapitre et de section, ainsi que la page. Comme ces références vont varier au fur et à mesure des modifications du texte, mieux vaut laisser LaTeX gérer ceci.

Lorsque l'on veut faire référence à un passage, on place une étiquette à l'endroit désiré, avec la commande `\label{`*`étiquette`*`}`. Pour indiquer la référence, on utilise :

- `\ref{`*`étiquette`*`}` pour mettre le numéro de chapitre et de section ;
- `\pageref{`*`étiquette`*`}` pour mettre le numéro page.

Ceci nécessite une double compilation :

- lors de la première compilation, LaTeX relève les emplacements des étiquettes et les place dans le fichier `.aux`, et génère des erreurs `Reference undefined` ;
- lors de la seconde compilation, il génère les références.

Exemple :

```
Les points forts de \LaTeX{} sont:\label{points-forts}
[…]
Comme relevé précédemment (section~\ref{points-forts} p.~\pageref{points-forts}), […]
```

Erreurs possibles

- `Label '[…]' multiply defined` : une étiquette a été utilisée à deux endroits différents ;
- `Reference '[…]' on page […] undefined` : un `\ref`/`\pageref` appelle une étiquette non définie ; vous avez probablement oublié de mettre une étiquette ou vous avez fait une faute de frappe entre le `\label` et le `\ref`/`\pageref` ;
- `Labels may have changed` : indique qu'il faut faire une autre compilation.

Bibliographie

Il est fréquent de devoir intégrer des références bibliographiques. Cela peut se faire sous la forme de notes de bas de page, en employant éventuellement les abréviations *ibid.*, *op. cit.* ou *loc. cit.* pour éviter les redondances.

Il est fréquent de regrouper les références à la fin du document. La réalisation « à la main » de cette tâche est vite pénible et source d'erreur, notamment si l'on change la disposition de l'ouvrage.

Heureusement, LaTeX dispose d'un outil auxiliaire appelé BibTeX (http://www.bibtex.org) [archive] qui gère automatiquement la bibliographie.

La description des ouvrages — auteurs, titre, année de parution, éditeur, … — sont écrites dans un fichier extérieur ; vous avez alors un fichier unique contenant toute votre bibliographie, et auquel se réfèrent vos différents documents LaTeX. Ainsi, si un ouvrage est mentionné dans plusieurs documents, il n'est rentré qu'une seule fois (écrire une fois, et lire beaucoup), ce qui limite les erreurs.

L'inclusion des référence bibliographiques dans le document LaTeX nécessite l'utilisation du programme `bibtex` sur le document LaTeX.

Vous devez donc créer un fichier de bibliographie. Il s'agit d'un fichier texte ayant l'extension `.bib`. Dans notre exemple, nous l'appelons `mabiblio.bib`.

L'extension `tocbibind` permet de faire figurer la bibliographie dans la table des matières.

Fichier de bibliographie

La constitution de ce fichier suit un formalisme précis :

- il faut choisir une référence unique pour l'ouvrage, elle sera utilisée dans le document LaTeX ; on utilise fréquemment les initiales de l'auteur ou des auteurs suivi de l'année, mais le choix est totalement libre ;
- on commence par indiquer le type de document (article, livre, thèse, actes d'une conférence, …) sous la forme d'un arobase suivi d'un mot-clef ; par exemple, `@book` pour un livre, `@article` pour un article, …
- la description de l'ouvrage est incluse dans un bloc `{…}` accolé au type précédent ; elle est de la forme *`champ = contenu`* ;
- les différents champs sont séparés par une virgule, et le contenu doit être entre guillemets droits `"…"` ou dans un bloc `{…}`.

Exemple

```
% ***** livres *****

@book{VER1875,
   author="Verne, Jules",
   title="Michel {Strogoff}",
   year="1875",
   publisher="Le livre de poche"
}

% ***** articles *****

@article{MERQUI2007,
   author="Merchet, Jean-Dominique and Quinot, Paul",
   title="L'{\'e}lection dans le miroir des sondages",
   journal="Lib{\'e}ration",
   number="21 f{\'e}vrier",
   year="2007"
}
```

On note que :

- certains mots sont entre accolade, ce qui permet de conserver leur mise en forme, notamment la casse ;
- il ne peut y avoir dans le fichier que des caractères sans diacritique (ni accent, ni cédille) ni ligature (pas de œ ni æ) ; pour utiliser de tels caractères, il faut donc utiliser les commandes `\'`, `\``, `\^`, `\"`, `\c`, … placées dans un bloc (par exemple `{\'e}` pour « é ») ;
- les auteurs sont sous la forme *`nom, prénom`*, et lorsqu'il y a plusieurs auteurs, ils sont séparés par le mot-clef `and`.

Document LaTeX

Lorsque l'on veut introduire la référence au livre, on place dans le texte `\cite{`*`mot_de_référence`*`}`. Par exemple

```
Ainsi, Jules \bsc{Verne} faisait dire à Wassili Fédor \cite{VER1875}:
```

Si l'on veut faire figurer un ouvrage dans la liste sans qu'il y ait de référence dans le texte, on utilise `\nocite`, n'importe où —mais à un endroit où on le retrouve facilement, par exemple juste avant la table des références—, par exemple

```
\nocite{VER1875}
```

À l'endroit où l'on veut placer la liste des ouvrages (en général à la fin), on met :

```
\bibliographystyle{style}
\bibliography{nom_du_fichier}
```

par exemple

```
\bibliographystyle{plain-fr}
\bibliography{mabiblio}
```

Le style `plain-fr` est un style francisé, c'est-à-dire que les noms de famille des auteurs sont en grandes et petites capitales et que les mots de description sont en français. Le style *plain* est un style dans lequel les références se font sous la forme d'un numéro entre crochets, les ouvrages étant numérotés dans l'ordre de citation dans le texte.

Compilation

Il faut compiler une première fois le document LaTeX avec `latex` afin qu'il repère les citations à la bibliographie. Puis, on effectue une compilation du document laTeX avec `bibtex`, et enfin une ou deux compilations avec `latex`.

Avec Texmaker

La compilation avec `bibtex` se fait avec la touche F11 ou bien avec le menu **Tools | BibTeX**.

Avec TeXShop

Pour compiler avec `bibtex` : sélectionner **BibTeX** dans le menu déroulant des programmes (le premier menu) et cliquer sur le bouton Composition ; puis, revenir sur l'option **LaTeX** du menu déroulant des programmes, et cliquer à nouveau sur Composition pour recompiler avec `latex`.

En ligne de commande

La compilation avec `bibtex` se fait par la commande `bibtex` *`nom_du_document_LaTeX`*.

Voir aussi

Sur Wikibooks

- LaTeX/Gestion de la bibliographie

Liens externes

- Bib-fr (http://tug.ctan.org/tex-archive/biblio/bibtex/contrib/bib-fr/) [archive] sur le TUG.

Mise en forme du texte

Cette section présente les instructions de mise en forme élémentaires, et comment créer des commandes personnelles permettant la séparation du fond et de la forme.

Choix de la police

La fonte est le dessin général des lettres, qui se décline ensuite en plusieurs corps (taille des lettres), formes (romain, italiques, petites capitales) et graisse (médium ou gras). L'ensemble fonte-corps--forme-graisse est une police. On parle aussi de famille de la police pour désigner la fonte.

Choix du style

Par défaut, on utilise les caractères dits « romains ». Pour mettre un texte en italiques, on peut utiliser les instructions suivantes :

- `\textit{`*`texte à mettre en italique`*`}` ;
- `{\itshape `*`texte à mettre en italique`*`}` ;
- `\itshape `*`texte à mettre en italique`*` \upshape` (up pour *upright*) ;
- `\begin{itshape}` *`texte à mettre en italique`* `\end{itshape}`.

Le plus simple semble `\textit{…}`. L'instruction `\itshape` change la forme du texte qui suit jusqu'à rencontre une autre instruction indiquant de remettre les lettres droites, `\upshape` ; ceci peut être utile dans la conception de commandes personnelles. Enfin, dans le deuxième exemple, on a inséré l'instruction `\itshape` à l'intérieur d'un bloc `{…}` ; le bloc limite la zone dans laquelle s'applique l'instruction.

L'avantage de l'instruction `\emph` sur `\textit` est double :

- il s'agit d'une indication de la fonction (emphase) et non de la mise en forme ; ainsi, si l'italique est utilisée pour différentes fonctions (mise en emphase, mot étranger, citation sans guillemets, …) on pourra aisément retrouver les portions de texte ayant telle ou telle fonction (au lieu de chercher parmi toutes les portions en italiques), il suffira de créer des commandes personnelles ayant le même effet, mais ayant un nom différent (voir ci-après) ;
- une emphase à l'intérieur de l'emphase est en romain.

Exemple

```
Voici \emph{un texte en emphase contenant une \emph{emphase}, intéressant, non~?}
```

Voici *un texte en emphase contenant une* emphase, *intéressant, non ?*

Pour mettre du texte en gras *(bold face)*, on peut utiliser :

- `\textbf{`*`texte à mettre en gras`*`}` ;
- `{\bfseries `*`texte à mettre en gras`*`}` ;
- `\bfseries `*`texte à mettre en gras`* `\mdseries` (md pour *medium*) ;
- `\begin{bfseries}` *`texte à mettre en gras`* `\end{bfseries}`.

Même remarques que ci-dessus.

Pour mettre du texte en petites capitales *(small capitals)*, on peut utiliser :

- `\textsc{`*`texte à mettre en petites capitales`*`}` ;
- `{\scshape `*`texte à mettre en petites capitales`*`}` ;
- `\scshape `*`texte à mettre en petites capitales`* `\upshape` ;
- `\begin{scshape}` *`texte à mettre en petites capitales`* `\end{scshape}`.

Même remarques que ci-dessus.

Dans la pratique, quasiment personne n'utilise les environnements (`\begin{…}` et `\end{…}`) pour les polices.

L'extension `babel` avec le paramètre `frenchb` fournit l'instruction `\bsc{…}` pour *boxed small capitals*, qui écrit le mot en petites capitales et empêche sa césure en fin de ligne (utile pour les noms propres par exemple).

La commande `\textnormal{…}` met le texte en police par défaut (minuscules romaines non grasses). On peut aussi utiliser `\normalfont`, par exemple

`\scshape … \normalfont` ou bien `{\scshape … {\normalfont … } …}`

Commande		Style
à argument	déclarative	
`\textnormal{...}`	`{\normalfont...}`	fonte du corps du document
`\emph{...}`	`{\em...}`	*emphase*
`\textup{...}`	`{\upshape...}`	forme droite
`\textit{...}`	`{\itshape...}`	*italique*
`\textsc{...}`	`{\scshape...}`	PETITES CAPITALES
`\textbf{...}`	`{\bfseries...}`	**gras**
`\textmd{...}`	`{\mdseries...}`	gaisse moyenne

On peut combiner les mises en forme, comme mettre un texte en italiques et gras, par exemple :

- `\textit{\textbf{texte à mettre en italique gras}}`
- `{\itshape \bfseries texte à mettre en italique gras}`.

Par contre, toutes les combinaisons ne sont pas possible ; en particulier, il n'existe pas de petites capitales en gras.

Citons enfin trois autre mises en forme :

- le soulignement : `\underline{texte à souligner}` ;
- les chiffres dits « bas de casse » ou « elzéviriens » (0123456789) : `\oldstylenums{chiffres}` ;
- les supérieures (ou texte en exposant) : `\textsuperscript{texte en position supérieurs}`.

Utilisation conventionnelle[1]

Nous ne considérons ici que le corps du texte, le reste — titres, légendes des figures, notes … — étant géré directement par LaTeX (on peut le configurer, mais cela sort cadre du présent chapitre).

Habituellement, on utilise essentiellement l'italique, et ce pour :

- mettre un ou plusieurs mots en emphase, mais on préfèrera l'usage des guillemets à la place ;
- indiquer un mot étranger, une locution latine ;
- indiquer une citation au sein d'un texte, on n'utilise alors pas de guillemets ;
- dans les parties de l'ouvrage qui ne sont pas de l'auteur : préface, avis de l'éditeur, …
- pour les titres d'ouvrages et d'œuvres (*Germinal* d'Émile Zola), le nom propre d'un navire (le paquebot *France*), les réalisation industrielles (le programme *Apollo*) , certaines créations commerciales (le *Numéro 5* de Chanel);
- les notes de musique.

Le gras n'est que rarement utilisé. Il permet de faire ressortir des mots du texte (mise en emphase), mais contrairement à l'italique, il attire l'attention au sein de la page. En mathématiques, il peut être utilisé pour indiquer les noms des ensembles (p.-ex. **N** pour les entiers naturels) lorsque l'on n'utilise pas les lettres ajourées (ℕ), ou pour indiquer les vecteurs (p.-ex. **v**) en typographie anglaise, lorsque l'on n'utilise pas les flèches ($\vec{v}$).

Les grandes capitales sont utilisées :

- pour les majuscules : aux initiales des noms propres, en début de phrase et pour les sigles (abréviations reprenant les initiales de mots[2]) ;
- sur les enveloppes postales en France, pour les noms des villes (norme AFNOR XPZ 10-011 *Spécifications postales - Adresse postale*, code postal français (http://www.laposte.fr/sna/rubrique.php3?id_rubrique=88) [archive]) ; dans ce cas, les capitales sont non accentuées ;
- pour les chiffres romains :
  - pour les millénaires ;
  - pour les divisions principales d'un livre : numéros d'actes d'une pièce de théâtre, annexes, psaume, fascicule, chanson, planche hors texte, …
  - pour les ans du calendrier républicain, les numéros de dynastie (Louis XIV).

Les petites capitales, quant à elles, sont utilisées :

- pour les noms de famille (p.-ex. Victor Hugo) ;
- pour le ou les premiers mots suivant une lettrine (grande lettre commençant un chapitre) ;
- pour les chiffres romains :
  - pour les siècles, en chiffres romains (p.-ex. XX^{e} siècle) ;
  - les divisions secondaires d'un ouvrage, à l'exception de *premier* et de *première* qui s'écrivent en entier (chapitre premier) : chapitre (chapitre II), scène d'une pièce de théâtre, couplet d'une chanson, épître, …

On aura par exemple « acte I scène III ».

Le soulignement n'a pas d'utilisation en typographie classique. Il est utilisé par l'auteur à la machine à écrire pour indiquer au typographe de mettre le texte souligné en italique, mais il n'est pas utilisé dans les livres ou journaux.

Le texte en supérieur est principalement utilisé pour les abréviations (1er, n°, …), et, en mathématiques, pour indiquer une élévation à une puissance ou bien un indice (composante covariante).

Choix de la fonte

LaTeX utilise par défaut des fontes appelées *extended computer modern* (EC). Vous avez bien sûr la possibilité de choisir d'autres fontes, mais cela

sort du cadre présent — une première approche simple. Indiquons simplement que l'extension `times` permet d'utiliser des fontes plus courantes (Times, Helvetica et Courier) à la place des fontes EC ; il faut alors mettes `\usepackage{times}` dans l'en-tête.

Par défaut, la fonte est une fonte avec empattement (ou *serif*). Vous pouvez utiliser à la place une fonte sans empattement *(sans serif)* :

- `\textsf{`*`texte à mettre en fonte sans empattement`*`}` ;
- `{\sffamily` *`texte à mettre en fonte sans empattement`* `}` ;
- `\sffamily` *`texte à mettre en fonte sans empattement`* `\rmfamily` (rm pour *roman*) ;
- `\begin{sffamily}` *`texte à mettre en fonte sans empattement`* `\end{sffamily}`.

Vous pouvez aussi utiliser une fonte de type machine à écrire *(teletype)*, à chasse fixe :

- `\texttt{`*`texte à mettre en fonte à chasse fixe`*`}` ;
- `{\ttfamily` *`texte à mettre en fonte à chasse fixe`* `}` ;
- `\ttfamily` *`texte à mettre en fonte à chasse fixe`* `\rmfamily` ;
- `\begin{ttfamily}` *`texte à mettre en fonte à chasse fixe`* `\end{ttfamily}`.

Commande		Style
à argument	déclarative	
`\textnormal{...}`	`{\normalfont...}`	fonte du corps du document
`\textrm{...}`	`{\rmfamily...}`	police romaine
`\textsf{...}`	`{\sffamily...}`	police linéale (sans sérif)
`\texttt{...}`	`{\ttfamily...}`	police machine à écrire

Utilisation conventionnelle[1]

Nous ne considérons ici que le corps du texte.

Habituellement, le corps du texte utilise une seule et unique fonte. Dans les ouvrages informatiques, on utilise souvent une fonte à chasse fixe pour représenter ce qui est entré au clavier ou ce qui apparaît à l'écran.

Choix du corps

Le corps général du texte est choisi lorsque l'on indique la classe du document (voir *Premier exemple* > *Amélioration*). LaTeX gère lui-même les variations de corps pour les titres, notes, …

On peut indiquer à LaTeX d'utiliser un corps plus grand ou plus petit :

- corps très petit :
  - `{\footnotesize` *`texte très petit`*`}`, ou bien
  - `\footnotesize` *`texte très petit`* `\normalsize`, ou bien
  - `\begin{footnotesize}` *`texte très petit`* `\end{footnotesize}` ;
- corps petit :
  - `{\small` *`texte petit`*`}`, ou bien
  - `\small` *`texte petit`* `\normalsize`, ou bien
  - `\begin{small}` *`texte petit`* `\end{small}` ;
- corps grand :
  - `{\large` *`texte grand`*`}`, ou bien
  - `\large` *`texte grand`* `\normalsize`, ou bien
  - `\begin{large}` *`texte grand`* `\end{large}` ;
- corps très grand :
  - `{\Large` *`texte très grand`*`}`, ou bien
  - `\Large` *`texte très grand`* `\normalsize`, ou bien
  - `\begin{Large}` *`texte très grand`* `\end{Large}`.

Utilisation conventionnelle[1]

Nous ne considérons ici que le corps du texte.

On utilise un corps plus petit (`\small`) pour du texte mis à l'écart du reste du texte (dans un paragraphe avec des marges plus grandes), comme par exemple dans un bloc de citation, ainsi que pour les épigraphes (ou exergue : citation en tête d'un livre ou au début d'un chapitre, en rapport avec son esprit).

On peut aussi utiliser la variation de corps comme effet esthétique (parangonage), mais avec parcimonie.

Exemple

```
Voici un texte avec
\emph{de l'italique},
\textbf{du gras},
\textsc{des petites capitales},
```

```
\textsf{des caractères sans empattement},
\texttt{des caractères à chasse fixe},
des mots avec {\small{un corps plus petit}} ou {\large{plus grand}}.
```

Voici un texte avec *de l'italique*, **du gras**, DES PETITES CAPITALES, des caractères sans empattement, `des caractères à chasse fixe`, des mots avec un corps plus petit ou plus grand

Composition du texte

Par défaut, en typographie française, le texte est composé en alinéa, c'est-à-dire qu'il est justifié (les lignes font toutes la même longueur), sauf pour la première ligne qui est rentrante (alinéa, ou indentation) et la dernière ligne qui est creuse (alignée à gauche).

On indique un nouvel alinéa, ou paragraphe, en laissant une ou plusieurs lignes vides.

On peut annuler l'alinéa en mettant la commande `\noindent` en début de paragraphe. On a alors une composition en pavé.

Au sein d'un alinéa, on peut faire un retour à la ligne en mettant deux contre-obliques « \\ ». C'est habituellement peu utilisé, mis à part dans les titres, ou dans une liste pour isoler un ou plusieurs mots dans un même item.

On peut aussi composer le texte en drapeau :

- drapeau droit ou composition au fer à gauche (texte aligné à gauche) : on débute le texte par `\begin{flushleft}` et on le termine par `\end{flushleft}` ;
- drapeau gauche ou au fer à droite (texte aligné à droite) : on débute le texte par `\begin{flushright}` et on le termine par `\end{flushright}`.

Enfin, on peut centrer le texte : on le débute par `\begin{center}` et on le termine par `\end{center}`.

Par contre, il n'existe pas de moyen simple de composer le texte en sommaire (justifié, mais avec la première ligne saillante), mise à part les listes (voir *Les environnements* > *Listes*).

Certaines citations longues sont mise en évidence en mettant :

- un blanc vertical avant et un blanc vertical après ;
- des marges plus grandes.

Pour cela, on commence le texte par `\begin{quotation}` et on le termine par `\end{quotation}`.

Si la citation n'a qu'un seul paragraphe et que l'on veut supprimer l'alinéa, on utilisera `\begin{quote}` et `\end{quote}`.

Exemple :

```
Si l'on considère ce passage de \emph{L'\'Ecole des femmes}~:

\small

\begin{center}
   \bsc{Chrysalde}
\end{center}

\begin{quote}
   Nous sommes ici seuls, et l'on peut, ce me semble, \\
   Sans craindre d'être ouïs y discourir ensemble. \\
   Voulez-vous qu'en ami je vous ouvre mon c\oe{}ur~? \\
   Votre dessein, pour vous, me fait trembler de peur~; \\
   Et de quelque façon que vous tourniez l'affaire, \\
   Prendre femme est à vous un coup bien téméraire.
\end{quote}

\normalsize
```

Si l'on considère ce passage de *L'École des femmes* :

CHRYSALDE

Nous sommes ici seuls, et l'on peut, ce me semble,
Sans craindre d'être ouïs y discourir ensemble.
Voulez-vous qu'en ami je vous ouvre mon cœur ?
Votre dessein, pour vous, me fait trembler de peur ;
Et de quelque façon que vous tourniez l'affaire,
Prendre femme est à vous un coup bien téméraire.

Commandes personnelles

Abréviation

Si l'on utilise fréquemment un terme long, comme par exemple le nom d'une molécule chimique ou un nom propre, on a intérêt à en créer une forme abrégée. Cela se fait avec la commande `\newcommand`, sous la forme :

```
\newcommand{\formeabrégée}{forme complète}
```

La forme abrégée commence par une contre-oblique — c'est une nouvelle instruction —, le terme ne doit pas être une instruction existante et ne doit comporter que des lettres : pas de signe de ponctuation ou d'espace, pas de caractère réservé, pas de chiffre.

On peut placer cette définition n'importe où avant que la commande soit utilisée, mais sa place « naturelle » est dans le préambule, avant le `\begin{document}` : cela permet de la retrouver facilement.

Par exemple :

```
préambule classique

\newcommand{\DND}{\emph{Donjons \& Dragons}\texttrademark}
\newcommand{\TNT}{trinitrotoluène}

\begin{document}

Dans \DND, on n'utilise pas de \TNT{} mais des boules de feu.

\end{document}
```

donne

Dans *Donjons & Dragons*™, on n'utilise pas de trinitrotoluène mais des boules de feu.

On évite ainsi de faire des fautes lorsque l'on écrit les termes.

On remarque que l'on retrouve le problème général des commandes en lettre : elles doivent être suivies d'un espace ou d'un signe de ponctuation pour indiquer leur fin.

Instruction de fonction

Pour indiquer la mise en forme pour une fonction particulière du texte, on crée une commande comme ci-dessus. Par exemple :

```
\newcommand{\langue}{\emph} % mots en langues étrangères
\newcommand{\cititalique}{\emph} % citation en italique
\newcommand{\nomprog}{\texttt}  % nom de programme en police teletype
```

Dans les cas complexes, il faut créer une commande avec paramètre, le paramètre étant le texte concerné :

```
\newcommand{\fonction}[1]{définition de la commande}
```

Le « [1] » indique qu'il n'y a qu'un seul paramètre, et celui-ci est désigné par « #1 » dans la définition Par exemple,

```
\newcommand{\citguill}[1]{\og #1 \fg} % citation entre guillemets
\newcommand{\important}[1]{\textit{\textbf{#1}}}
\newcommand{\Isiecle}{\textsc{i}\ier}
\newcommand{\siecle}[1]{\textsc{#1}\ieme}

\important{Attention~!} Ceci n'est vrai que du \Isiecle{} au \siecle{iii} siècle.
```

donne

Attention ! Ceci n'est vrai que du Ier au IIIe siècle.

On trouvera une solution plus élégante dans Bitouzé et Charpentier[3] p. 262.

Erreurs possibles

Si le nom que l'on utilise pour la nouvelle commande est déjà utilisé, la compilation génère une erreur

```
! Command name … already defined.
```

Dans ce cas, il faut changer le nom de la commande. Si vous voulez redéfinir une commande existante, il faut utiliser l'instruction `\renewcommand`, mais attention aux effets indésirables…

Préambule général

La plupart des documents que vous créez ont le même préambule : vous faites en général appel aux mêmes extensions et utilisez les mêmes commandes personnelles. Vous pouvez donc créer un fichier `.tex` ne contenant que les données du préambule, et l'invoquer au début de vos documents ; vous avez ainsi une « bibliothèque de commandes » commune. Si par exemple vous appelez cette bibiothèque `preambule.tex`, vous commencerez vos documents par

```
\documentclass[options]{classe}

\input{preambule.tex}

\begin{document}
…
```

Ainsi, si vous créez une nouvelle commande personnelle, il suffit de modifier le fichier de préambule pour en faire profiter tous vos documents. De même, si vous modifiez une commande personnelle dans le fichier de préambule, il suffit de recompiler vos documents pour que cette modificaiton soit prise en compte.

On est donc un niveau au dessus en matière de séparation du fond et de la forme…

Vous pouvez également avoir plusieurs fichiers de préambule, un par famille de document.

Notez que l'on peut également créer sa propre classe (fichier `.cls` appelé par `\documentclass`) ou extension (fichier `.sty` appelé par `\usepackage`), mais cela nécessite d'apprendre quelques commandes supplémentaires, ce qui est inutile si l'on veut juste se contenter d'un préambule général.

Notes

1. *Lexique des règles typographiques en usage à l'Imprimerie nationale*, éd. Imprimerie nationale, 2002, ISBN 2-7433-0482-0
2. lorsque le sigle se prononce comme un mot au lieu de s'épeler (sigle lexicalisé), on parle d'acronyme et on l'écrit en général en bas de casse (minuscules), comme « laser », mais il s'agit là d'un usage, pas d'une règle générale, on écrit par exemple CEDEX, CNES, ONU
3. D. Bitouzé et J.-C. Charpentier, *LaTeX, synthèse et cours*, éd. Pearson Education, 2006, ISBN 2-7440-7187-0

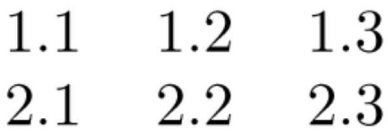
Faire des tableaux

N. B. : Pour obtenir des tableaux typographiquement corrects, il faut charger l'extension `array`. *On placera donc un* `\usepackage{array}` *dans le préambule du document. Cependant, sauf exception signalée, les exemples ci-dessous sont utilisables sans cette extension.*

Commencons par faire un tableau assez simple qui contient 2 lignes et trois colonnes :

Fichier source **Résultat (en PostScript)**

```
\begin{tabular}{lll}
   1.1 & 1.2 & 1.3 \\
   2.1 & 2.2 & 2.3 \\
\end{tabular}
```



L'environnement de base pour les tableaux en LaTeX est `tabular`, il prend un argument obligatoire qui spécifie le nombre de colonnes du tableau :

```
\begin{tabular}{<colonnes>}
   <lignes>
\end{tabular}
```

L'argument <*colonnes*> est une suite de caractères qui peuvent être :

- un `l` pour une colonne alignée à gauche *(left)* ;
- un `r` pour une colonne alignée à droite *(right)* ;
- un `c` pour une colonne centrée *(center)*.
- un `p{`*`largeur`*`}` pour une colonne avec un paragraphe en pavé (c'est-à-dire justifié, mais sans alinéa) ; la largeur est indiquée de manière classique, par exemple `p{3cm}` ou `p{10em}`.

Les <*lignes*> sont représentées par les valeurs des colonnes séparées par des esperluettes `&` et terminées par deux contre-obliques `\\`.

Si l'on veut séparer les colonnes par des filets (traits) verticaux, on ajoutera des tubes `|` à l'endroit souhaité dans l'argument de l'environnement.

On peut aussi ajouter des filets horizontaux avec la commande `\hline`.

Fichier source **Résultat (en PostScript)**

```
\begin{tabular}{|l|c|r|}
  \hline
  colonne 1 & colonne 2 & colonne 3 \\
  \hline
  1.1 & 1.2 & 1.3 \\
  2.1 & 2.2 & 2.3 \\
  \hline
\end{tabular}
```

colonne 1	colonne 2	colonne 3
1.1	1.2	1.3
2.1	2.2	2.3

On peut faire suivre deux `\hline` pour avoir une ligne double.

> **Remarque :** même lorsque l'on n'a pas besoin de fonctionnalités spécifiques, il est utile de déclarer l'extension `array`, qui règle un certain nombre de défauts.

Fusionner les colonnes

Pour fusionner des colonnes, on utilise l'instruction `\multicolumn`. Celle-ci se place à l'endroit où se trouve la première cellule. Sa syntaxe est :

```
\multicolumn{<nombre>}{<colonne>}{<texte>}
```

où

- <*nombre*> est le nombre de colonnes fusionnées ;
- <*colonne*> indique l'alignement (`l`, `c`, `r`) ; le filet de gauche est le filet normal de la première cellule (celui défini dans la définition du tableau), mais il faut définir le filet de droite.

Par exemple :

```
\begin{tabular}{|l|c|r|}
   \hline
   colonne 1 & \multicolumn{2}{c|}{colonnes 2 \& 3} \\
   \hline
   1.1 & 1.2 & 1.3 \\
   \hline
   2.1 & 2.2 & 2.3 \\
   \hline
\end{tabular}
```

donne

colonne 1	colonnes 2 & 3	
1.1	1.2	1.3
2.1	2.2	2.3

Si l'on veut changer la composition du texte (alignement) pour une cellule, on peut utiliser `\multicolumn{1}{<colonne>}{<texte>}`. À moins que la première colonne ne soit incluse, on n'indique que le filet de droite (par exemple `{l|}`, le filet de gauche étant déterminé par la description générale.

Fusionner les lignes

La fusion des lignes *stricto sensu* nécessite le recours à une extension particulière, `multirow`. Son utilisation est décrite dans la section *Tableaux > Colonnes enveloppant des lignes multiples*. Nous allons ici procéder à une approche simplifiée, consistant à laisser des cellules vides et à tracer un filet horizontal ne faisant pas toute la largeur.

Pour tracer ce filet, on utilise l'instruction

```
\cline{<lign1>-<lign2>}
```

Par exemple :

```
\begin{tabular}{|l|c|r|}
   \hline
    colonne 1 & \multicolumn{2}{c|}{colonnes 2 \& 3} \\
    \hline
    1.1 & 1.2 & 1.3 \\
    \cline{2-3}
        & 2.2 & 2.3 \\
    \hline
\end{tabular}
```

donne

colonne 1	colonnes 2 & 3	
1.1	1.2	1.3
	2.2	2.3

Tableau flottant

On peut laisser notre typographe virtuel placer le tableau ; on parle alors de tableau flottant. On indique l'endroit où l'on souhaiterait voir le tableau (à côté du texte le précédant dans le fichier source, ou bien en haut ou en bas d'une page), et LaTeX fait de son mieux en fonction des contraintes, notamment de la présence d'autres objets flottants. S'il n'arrive pas à placer le flottant sur la page en cours, il est « mis en attente » et sera placé plus loin. De plus, LaTeX numérote automatiquement le tableau, ce qui permet de dresser un index des tableaux.

Pour cela, on encapsule le tableau dans un environnement `table` :

```
\begin{table}[<position>]
   \begin{tabular}{<colonnes>}
       […]
   \end{tabular}
\end{table}
```

où <*position*> est une lettre indiquant l'emplacement désiré :

- `h` pour qu'il soit à côté du texte précédant dans le source *(here)*,
- `t` : en haut d'une page *(top)*,
- `b` : en bas d'une page *(bottom)*,

- `p` : dans une page ne contenant que des flottants (regroupement des figures et tableaux).

Si l'on veut donner un titre et placer une étiquette permettant de faire référence au tableau (cf. *Structuration du texte* > *Références*), on utilisera la syntaxe suivante :

```
\begin{table}[<position>]
   \caption{\label{<étiquette>} <titre>}
   \begin{tabular}{<colonnes>}
      […]
   \end{tabular}
\end{table}
[…]
Dans le tableau~\ref{<étiquette>} page~\pageref{<étiquette>}, […].
```

Si l'on veut centrer le tableau dans l'environnement, il est recommandé d'utiliser l'extension `array` et la commande `\centering` plutôt que l'environnement `center` :

```
\usepackage{array}
[…]
\begin{table}[<position>]
   \centering
   \begin{tabular}{<colonnes>}
      […]
   \end{tabular}
\end{table}
```

Lorsqu'il y a trop de flottants, la mise en page peut devenir problématique. On pourra avoir recours à l'instruction `\clearpage` qui provoque un changement de page et l'affichage de tous les flottants en attente. L'instruction `\cleardoublepage` a le même effet, mais le texte qui suit est placé sur une page impaire (« belle page » si l'on est en recto-verso).

L'index des tables s'obtient avec l'instruction `\listoftables`. L'extension `tocbibind` permet de faire figurer cet index dans la table des matières.

Creation des tableaux en WYSIWYG

En ligne

- http://www.tablesgenerator.com
- http://truben.no/table

Logiciels

- La Table (Win & Mac) (http://www.ctan.org/pkg/latable) [archive]

Voir aussi

- Programmation LaTeX Tableaux

Inclure des images

L'inclusion d'images nécessite l'utilisation de l'extension `graphicx` (`\usepackage{graphicx}`).

Pour inclure une image, il faut utiliser la commande `\includegraphics{`*`nom du fichier`*`}`. On peut distinguer deux cas :

- si vous compilez avec `latex`, vous ne pouvez inclure que des images PostScript (avec l'extension `.ps` ou `.eps`) ;
- si vous compilez avec `pdflatex`, vous ne pouvez inclure que des images PNG (extension `.png`), JPEG (extension `.jpg` ou `.jpeg`) ou des fichiers PDF (extension `.pdf`) mais pas des images PostScript ; les images `.bmp` sont intégrables mais il faut en définir la hauteur et la largeur.

Vous pouvez omettre l'extension du fichier ; ainsi, vous pouvez avoir le fichier dans plusieurs formats, et laisser le compilateur choisir le fichier qui lui convient. On a donc typiquement deux situations :

- vous disposez d'une image de type matricielle *(bitmap)*, habituellement au format JPEG (photographie) ou PNG (dessin) : il faut convertir cette image à un format PostScript, vous pouvez pour cela l'ouvrir avec un logiciel de traitement d'image (comme `The Gimp`) et l'enregistrer sous le format EPS ; vous avez donc deux fichiers portant le même nom mais ayant une extension différente (un JPEG et un EPS, ou bien un PNG et un EPS) ;
- vous disposez d'une image vectorielle, habituellement au format SVG (par exemple créée par `Inkscape`) : il faut la sauvegarder au format PDF et au format EPS, les deux formats sont vectoriels, il n'y a donc pas de perte de qualité.

Chemin d'accès

Par défaut, LaTeX va chercher l'image dans le répertoire (dossier) où se trouve le fichier `.tex`. On peut indiquer un répertoire différent en utilisant la syntaxe Unix : les répertoires sont séparés par des barres de fraction `/`.

Par exemple, si le projet devient imposant, il est intéressant de placer les images dans un sous-répertoire, appelons-le `illustrations`. On peut alors :

- soit mettre le chemin d'accès dans la commande `\includegraphics` :
  `\includegraphics{illustrations/image}` ;
- soit définir le chemin d'accès dans le préambule avec une commande `\graphicspath` :
  `\graphicspath{{illustrations/}}`
  …
  `\includegraphics{image}`.

On remarque que :

- à l'intérieur de `\graphicspath{…}`, le chemin d'accès est entre accolades ;
- le chemin d'accès se termine par une barre de fraction `/`.

Taille de l'image

L'idéal est d'avoir une image directement au format dans lequel elle doit être incluse. Il faudra faire attention à la notion de résolution : le rendu peut être très différent entre l'affichage à l'écran, la sortie sur imprimante et l'impression *offset*.

Si l'on veut retailler l'image, le mieux est d'utiliser les options de `\includegraphics` fournies par l'extension `graphicx` :

- `\includegraphics[width=`*`largeur`*`]{`*`nom du fichier`*`}` pour fixer la largeur ;
- `\includegraphics[height=`*`hauteur`*`]{`*`nom du fichier`*`}` pour fixer la hauteur ;
- `\includegraphics[scale=`*`échelle`*`]{`*`nom du fichier`*`}` pour fixer l'échelle.

Rappelons que les paramètres *hauteur* et *largeur* sont composés d'un nombre avec une unité accolée, par exemple `10cm` pour dix centimètres. Le paramètre *échelle* quant à lui n'a pas d'unité :

- `2` double la largeur (par exemple agrandissement A5 → A3).
- `1.4` double la surface (par exemple agrandissement A4 → A3) ;
- `1` désigne la taille réelle ;
- `0.7` réduit la surface de moitié (par exemple réduction A3 → A4) ;
- `0.5` réduit la largeur de moitié (par exemple réduction A3 → A5).

D'autres options sont disponibles, voir *Images > Extension graphicx*.

Encadrement

On peut inclure l'image dans un cadre en utilisant la fonction `\fbox` (cf. *Options de mise en forme avancées > Cadre et couleur du fond*).

Figure flottante

On peut laisser notre typographe virtuel placer l'image ; on parle alors de figure flottante. Le procédé est similaire aux tableaux flottants (cf. *Faire*

des tableaux > Tableau flottant).

Pour cela, on encapsule l'image dans un environnement `figure` :

```
\begin{figure}[position]
   \includegraphics[…]{…}
\end{figure}
```

où

- *position* est une lettre indiquant l'emplacement désiré :
  - `h` pour qu'il soit à côté du texte précédant dans le source *(here)*,
  - `t` : en haut d'une page *(top)*,
  - `b` : en bas d'une page *(bottom)*,
  - `p` : dans une page ne contenant que des flottants (regroupement des figures et tableaux).

Latex tient compte des règles internes de mise en page (une image par page ou toutes les images en haut d'une page par exemple) en priorité pour positionner une image. Pour forcer l'emplacement d'une image il faut faire précéder la lettre de *position* par un `!` (par exemple `[!h]`).

Si l'on veut placer une légende ainsi qu'une étiquette permettant de faire référence à l'image (cf. *Structuration du texte > Références*), on utilisera la syntaxe suivante :

```
\begin{figure}[position]
   \caption{\label{étiquette} titre}
   \includegraphics[…]{…}
\end{figure}
…
Dans la figure~\ref{étiquette} page~\pageref{étiquette}, …
```

Lorsqu'il y a trop de flottants, la mise en page peut devenir problématique. On pourra avoir recours à l'instruction `\clearpage` qui provoque un changement de page et l'affichage de tous les flottants en attente. L'instruction `\cleardoublepage` a le même effet, mais le texte qui suit est placé sur une page impaire (« belle page » si l'on est en recto-verso).

L'index des figures s'obtient avec l'instruction `\listoffigures`. L'extension `tocbibind` permet de faire figurer cet index dans la table des matières.

Dessiner avec LaTeX

LaTeX propose des instructions pour faire des dessins, ou tracer des graphiques. Outre l'environnement `picture`, vous disposez des extensions de la suite PSTricks, qui permet également de tracer des fonctions mais doit être compilé avec `latex` et non pas avec `pdftex` (on peut obtenir un fichier PDF à partir du fichier PostScript généré), ou avec PGF/TikZ.

Cela dépasse largement le cadre de ces *premiers pas*. LaTeX n'est pas un logiciel de dessin, et s'il est possible de dessiner, avec un rendu magnifique et en générant un fichier très petit, cela requiert l'apprentissage d'instructions supplémentaires.

Il existe cependant des programmes de dessin qui génèrent du code PSTricks. On peut ainsi générer un fichier LaTeX et l'inclure dans le fichier en cours, pour générer une image de grande qualité pour une taille modeste (toutefois, le code généré n'est pas forcément très lisible ni optimal).

Citons par exemple TeXgraph de Patrick Fradin :

- page officielle TeXgraph (http://texgraph.tuxfamily.org/index.html) [archive] ;
- forum TeXgraph (http://texgraph.tuxfamily.org/forum/) [archive].

> *Pour plus de détails voir :* ***LaTeX/Dessiner avec LaTeX***.

Voir aussi

Sur Wikipédia

- Joint Photographic Experts Group
- Portable Network Graphics
- PostScript

Ailleurs

- Using Imported Graphics in LaTeX and pdfLaTeX (http://www.ctan.org/tex-archive/info/epslatex.pdf) [archive] (fichier PDF, 988 Ko, 124 p)

texte - Choix de la police - Mise en page - Mathématiques - Gestion des gros documents - Faire des présentations - Arts et loisirs - Dessiner avec LaTeX - Créer une extension ou une classe - Programmer avec LaTeX - *Annexes* - Vade mecum - Conversion - Glossaire de typographie - Index - Commandes - Liens externes

Options de mise en forme avancées

Mise en forme du texte

Césure

La césure (coupure d'un mot en fin de ligne afin de respecter la justification et le gris typographique) est gérée automatiquement par LaTeX. En particulier, le respect des règles françaises est assuré par l'utilisation de l'option T1 de l'extension `fontenc`. Il peut toutefois arriver que la césure ne soit pas correcte ; par exemple, la césure modifie la prononciation, ou bien ne respecte pas l'étymologie (notamment dans le cas d'un mot composé).

Pour indiquer l'endroit où peut avoir lieu la césure, on utilise la commande `\hyphenation{`*`liste de mots`*`}` dans le préambule. Elle contient une liste de mots, séparés avec des espaces, et contenant un tiret « - » pour indiquer les endroits où l'on peut couper, par exemple

```
\hyphenation{anti-consti-tu-tion-nel-le-ment atmo-sphère caou-tchouc cis-alpin trans-action}
```

Si l'on veut empêcher une césure, il suffit de mettre le mot à l'intérieur d'un `\mbox{…}`, par exemple

```
Le roi \mbox{Nabuchodonosor} régnait…
```

Notons que la commande `\bsc{…}` de l'extension `babel/frenchb`, utilisée pour les patronymes des auteurs, interdit la césure.

Lettrine

On commence souvent un chapitre en belle page par une lettrine : la première lettre est au fer à gauche, et avec un corps plus grand (en général sur deux lignes), et le ou les mots suivants sont en petite capitale.

On dispose pour cela de l'extension `lettrine`, qui s'utilise comme suit :

```
\usepackage{lettrine}
[…]
\lettrine{L}{es premiers mots} du premier paragraphe ''[…]''
```

Le « L » est alors en capitale de grand corps, et « es premiers mots » est en petites capitales.

Si l'on veut avoir une lettrine en gothique, on peut faire :

```
\usepackage{lettrine}
\usepackage{oldgerm}

[…]
\lettrine{\textgoth{L}}{es premiers mots} du premier paragraphe […]
```

On peut aussi avoir recours à la famille `initfamily` de l'extension `yfonts` pour avoir des lettrines enluminées :

```
\usepackage{lettrine}
\usepackage{yfonts}

\newcommand{\enluminure}[2]{\lettrine[lines=3]{\small \initfamily #1}{#2}}

[…]
\enluminure{L}{es premiers mots} du premier paragraphe […]
```

Voir aussi Choix de la police.

Couleur

Dans les instructions ci-dessous, le paramètre *couleur* est le nom de la couleur en anglais (sur fond blanc et sur fond noir, afin de rendre compte de la lisibilité) :

- `red` : rouge, rouge ;
- `green` : vert, vert ;
- `blue` : bleu, bleu ;
- `cyan` : cyan, cyan (bleu primaire) ;
- `magenta` : magenta, magenta (rose primaire) ;
- `yellow` : jaune, jaune ;
- `orange` : orange, orange ;
- `violet` : violet, violet ;
- `purple` : pourpre, pourpre ;

- `brown` : brun, brun ;
- `black` : noir, noir ;
- `darkgray` : gris sombre, gris sombre ;
- `gray` : gris, gris ;
- `lightgray` : gris clair, gris clair ;
- `white` : blanc, blanc.

Couleur des caractères

Pour mettre des caractères en couleur, il faut avoir recours à l'extension `xcolor` (http://www.ukern.de/tex/xcolor.html) [archive]. On utilise alors les instructions suivantes pour mettre des caractères de couleur :

- `\textcolor{`*`couleur`*`}{`*`texte`*`}` ;
- `{\color{`*`couleur`*`} `*`texte`*`}`.

Utilisation conventionnelle

Il n'y a pas d'utilisation conventionnelle des couleurs. Attirons toutefois l'attention sur :

- la lisibilité : un contraste insuffisant rend difficile la lecture, un contraste trop violent (comme jaune et bleu) la rend désagréable ;
- l'étalonnage des couleurs : la couleur à l'écran n'est pas celle qui est imprimée par une imprimante à jet d'encre, elle-même différente de celle d'une imprimante laser couleur ou d'une impression *offset* ;
- le rendu si l'on imprime en noir et blanc ou si l'on photocopie ;
- le vieillissement du document : la couleur des imprimantes à jet d'encre a tendance à pâlir avec le temps ;
- la couleur permet d'attirer l'attention sur un point ; trop de couleurs fait perdre le fil de la lecture, on peut raisonnablement se limiter à quatre couleurs ;
- prendre en compte le coût de l'impression.

> *Pour plus de détails voir :* ***Rédaction technique/De l'usage des couleurs dans un document***.

Cadre et couleur du fond

La création d'un cadre se fait avec l'instruction `\fbox` *(framed box)* :

```
\fbox{texte à encadrer}
```

mais telle quelle, cette fonction garde tout sur la même ligne, au risque de dépasser de la page…

Si l'on veut avoir un texte encadré sur plusieurs lignes, par exemple un pavé, il faut utiliser une minipage, avec l'environnement du même nom :

```
\fbox{\begin{minipage}{''largeur''}
''texte''
\end{minipage}}
```

où le paramètre *largeur* est exprimé dans les unités conventionnelles (cf. *Éléments de base > Espaces et changements de ligne*). Pour *largeur*, on peut aussi utiliser `\textwidth` pour avoir la largeur totale du texte (justification), ou bien `\linewidth` pour avoir la largeur de la ligne (qui peut être plus petite selon l'environnement courant). On peut aussi mettre un multiplicateur devant `\textwidth`, par exemple

```
\fbox{\begin{minipage}{0.9\textwidth}
   ''texte''
\end{minipage}}
```

Pour avoir une boîte faisant 90 % de la justification. Si l'on veut organiser le texte du code en mettant `\begin{minipage}` sur la ligne suivante, il faut mettre la fin de la ligne du `\fbox` en commentaire afin que LaTeX ne prenne pas en compte le caractère de fin de ligne

```
\fbox{%
   \begin{minipage}{0.9\textwidth}
      ''texte''
   \end{minipage}%
}
```

Si le texte à encadrer s'étend sur plusieurs pages, on aura recours à l'environnement `framed` de l'extension du même nom, ou bien à l'environnement `breakbox` de l'extension `eclbkbox` [13] (http://www.grappa.univ-lille3.fr/FAQ-LaTeX/6.20.html).

Pour avoir un fond coloré, il faut utiliser l'extension `xcolor`. On a alors :

- `\colorbox{`*`couleur`*`}{`*`texte`*`}` : *couleur* prend les mêmes valeurs qu'avec `\textcolor` ci-dessus, et la commande réagit comme `\fbox` (il faut

une minipage si le texte dépase la ligne) ;
- `\fcolorbox{`*`couleur cadre`*`}{`*`couleur fond`*`}{`*`texte`*`}` : crée une boîte avec un fond et un filet de couleur.

On peut également définir la couleur de fond de toute la page avec `\pagecolor{`*`couleur`*`}`.

Utilisation conventionnelle

Mêmes remarques que ci-dessus.

Modèles de couleur

On peut utiliser d'autres couleurs que celles prédéfinies. On dispose pour cela de plusieurs modèles :

- modèles de type rouge-vert-bleu (RVB) :
  - `rgb` : les trois composantes sont des nombres décimaux compris entre 0 et 1, séparés par une virgule,
  - `RGB` : les trois composantes sont des nombres entiers compris entre 0 et 255, séparés par une virgule,
  - `HTML` : les trois composantes sont des nombre hexadecimal entiers compris entre 00 et FF, accolés ;
- modèle cyan-magenta-jaune-noir : `cmyk`, les quatre composantes sont des nombres décimaux compris entre 0 et 1, séparés par une virgule ;
- modèle teinte-saturation-luminosité (TSL) : `hsb`, les trois composantes sont des nombres décimaux compris entre 0 et 1 (la teinte est donc l'angle en degrés divisé par 360), séparés par une virgule ;
- modèle nuances de gris : `gray`, avec un nombre décimal compris entre 0 (noir) et 1 (blanc).

Sans la syntaxe, on remplace alors {*couleur*} par [*modèle*]{*couleur*}, par exemple

```
\pagecolor[''modèle'']{''couleur''}
```

Voci plusieurs manière de définir le vert olive :

- `\textcolor[rgb]{0.5,0.5,0}{texte vert olive}`
- `\textcolor[RGB]{128,128,0}{texte vert olive}`
- `\textcolor[HTML]{808000}{texte vert olive}`
- `\textcolor[hsb]{0.17,1,0.5}{texte vert olive}`
- `\textcolor[cmyk]{0,0.17,0.67,0.33}{texte vert olive}` (la variante proposée ici est légèrement différente des précédentes)

Pour le gris :

- `\textcolor[gray]{0.5}{texte gris}`

Mais dans la démarche de la séparation du fond et de la forme, il vaut mieux définir une nouvelle couleur :

```
\definecolor{''nom''}{''modèle''}{''couleur''}
```

par exemple

```
\definecolor{vertolive}{rgb}{0.5,0.5,0}
\textcolor{vertolive}{texte vert olive}
```

Pour plus de détails voir : ***Rédaction technique/De l'usage des couleurs dans un document#Codage informatique***.

Interligne

L'extension `setspace` (basé sur l'ancienne extension `doublespace`) permet d'augmenter l'interlignage, en introduisant l'environnement `spacing` :

```
\begin{spacing}{''facteur''}
   […]
\end{spacing}
```

multiplie l'interligne du *facteur* indiqué. Par exemple pour avoir un interligne de 1,2 fois l'interligne normal :

```
…
\usepackage{setspace}
…
\begin{document}
\begin{spacing}{1.2}
…
\end{spacing}
\end{document}
```

Pour les interlignes de une fois et demie et double, on dispose des commandes respectives `\onehalfspacing` et `\doublespacing` à placer dans le préambule.

On dispose également des environnements `singlespace`, `onehalfspace` et `doublespace`.

Texte en colonnes

Il est possible de présenter tout le texte en deux colonnes. Pour cela, on utilise l'argument `twocolumn` lors de l'appel de la classe, par exemple :

```
\documentclass[11pt, a4paper, twocolumn]{article}
```

On peut changer la disposition d'une page à l'autre :

- `\twocolumn` commence une nouvelle page en deux colonnes ;
- `\onecolumn` commence une nouvelle page en une colonne.

On peut utiliser un paramètre optionnel pour mettre un texte en une seule colonne au début d'une page à deux colonnes, comme par exemple un titre :

```
\twocolumn[\chapter{''titre''}]
```

Les versions étoilées des environnements flottants `table*` et `figure*` permettent de placer les objets flottants sur la largeur de la page au lieu de la largeur d'une colonne.

Si l'on désire utiliser plus de colonnes, ou si l'on désire changer le nombre de colonne en cours de page, on utilise alors l'extension `multicol` (sans *s*). Cela permet d'utiliser l'environnement `multicols` (avec un *s*), avec la syntaxe :

```
…
\usepackage{multicol}
…
\begin{document}
…
\begin{multicols}{''nombre de colonnes''}
   ''texte''
\end{multicols}
\end{document}
```

La fonction `\columnbreak` permet de forcer le passage à la colonne suivante.

Note

La présentation en multicolonne est adapté aux corps de petite taille (petite taille de texte), donc aux documents « compacts » : cela permet de limiter le nombre de caractères à lire avant de revenir à la ligne. Cela pose cependant le problème de la continuité de lecture lorsque l'on rencontre un intertitre ou une illustration : faut-il poursuivre la lecture sous l'intertitre/la figure, ou bien faut-il revenir en haut de la colonne suivante ?

Travailler avec deux langues

Si l'on a un document en plusieurs langues, on peut faire varier la marche typographique utilisée selon la langue du passage. Notez que dans le cas d'une citation en langue étrangère, on utilise la marche de la langue du document ; par exemple, si l'on a une citation en anglais dans un ouvrage en français, on garde la typographie française.

Pour utiliser plusieurs langues, il suffit d'indiquer plusieurs options à `babel` ; la dernière option est la langue par défaut. Puis, pour changer de langue, il suffit d'indiquer `\selectlanguage{`*`langue`*`}`. Par exemple, pour un texte en français avec des passages en anglais :

```
…
\usepackage[english, frenchb]{babel}
…
\begin{document}
…
\selectlanguage{english} % début du passage en typographie anglaise
  …
\selectlanguage{frenchb} % retour à la typographie française
…
\end{document}
```

Ajout d'un sommaire

Dans la typographie française, la table des matières se met à la fin d'un ouvrage. Il est alors fréquent d'avoir un sommaire en début d'ouvrage, c'est-à-dire une table des matières ne reprenant que les titres de chapitre.

Le problème est que la commande `\tableofcontents` ne peut être utilisée qu'une seule fois.

On peut utiliser la solution suivante [14] (http://groups.google.com/group/fr.comp.text.tex/browse_thread/thread/b29a38a820aac28c/443f3855c680b6c3?) :

```
\usepackage[tight]{shorttoc}

\newcommand{\sommaire}{\shorttoc{Sommaire}{1}}

\begin{document}

[…]

\sommaire

[…]

\tableofcontents

\end{document}
```

De manière générale, la commande `\shorttoc` utilise la syntaxe

```
\shorttoc{titre de la table}{profondeur}
```

où *profondeur* est le niveau de détail.

Les environnements

Un environnement est une zone de texte délimitée par deux balises

```
\begin{environnement}
   […]
\end{environnement}
```

Un environnement ouvert par un `\begin` doit toujours être fermé par un `\end`. Le document LaTeX est un environnement en soi, l'environnement `document`.

Principaux environnements

Nous récapitulons ci-après les principaux environnements déjà vus.

Alignements
: cf. *Mise en forme du texte > Composition du texte* :
- `flushleft` pour une composition en drapeau au fer à gauche (ou texte aligné à gauche) ;
- `flushright` pour une composition en drapeau au fer à droite (ou texte aligné à droite) ;
- `center` pour du texte centré.

Dans chacun des ces environnements, la commande `\\` permet de forcer un retour à la ligne.

Citations
: Pour les citations séparées du texte, on dispose des environnements `quote` et `quotation` ; cf. *Mise en forme du texte > Composition du texte*.

Listes
: Les listes structurées s'obtiennent également avec des environnements : `enumerate`, `itemize` et `description` ; cf. *Structuration du texte > Listes structurées*.

Minipage
: L'environnement `minipage` permet de considérer une portion de texte comme une page à part (qui peut même contenir des notes de bas de page) ; cf. *Cadre et couleur du fond* ci-dessus.

Objets flottants
: Les objets flottants sont dans des environnements :
- `table` pour les tableaux, cf. *Faire des tableaux > Tableau flottant* ;
- `figure` pour les images, cf. *Inclure des images > Figure flottante*.

Tableaux
: Les tableaux utilisent l'environnement `tabular`. Celui-ci est décrit dans la section *Faire des tableaux*.

Commandes personnelles

La définition d'une commande personnelle de type environnement `\begin{…}` et `\end{…}` (environnement) est un peu plus complexe. Si l'on veut reprendre l'exemple des citations précédent, on utilisera :

```
\newcommand{\titre}{\emph}
\newcommand{\personnage}[1]{\begin{center} \textsc{#1} \end{center}}
\newenvironment{tirade}
   {\begin{quote} \small}
   {\normalsize \end{quote}}

Si l'on considère ce passage de \titre{L'\'Ecole de femmes}~:
{\small \personnage{Chrysalde}  }

\begin{tirade}
   Nous sommes ici seuls, et l'on peut, ce me semble, \\
   Sans craindre d'être ouïs y discourir ensemble. \\
   Voulez-vous qu'en ami je vous ouvre mon c\oe{}ur~? \\
   Votre dessein, pour vous, me fait trembler de peur~; \\
   Et de quelque façon que vous tourniez l'affaire, \\
   Prendre femme est à vous un coup bien téméraire.
\end{tirade}
```

De manière générale, on a `\newenvironment{`*`nom de l'environnement`*`}{`*`instructions de début`*`}{`*`instructions de fin`*`}`. On pourra se reporter à Bitouzé et Charpentier[1] p. 263–254 et Roland[2] p. 139–140.

Erreurs possibles

- `! \begin{…} ended by \end{…}` : vous avez oublié de fermer un environnement, ou bien vous avez fait une faute en tapant le nom de l'environnement dans le `\end{…}`, ou bien vous avez essayé de faire se chevaucher des environnements, ce qui n'est pas possible ;
- `! command … already defined` : le nom que l'on utilise pour le nouvel environnement est déjà utilisé ; il faut changer le nom de l'environnement ; si vous voulez redéfinir un environnement existant, il faut utiliser l'instruction `\renewenvironment`, mais attention aux effets indésirables…
- `! Environment … undefined` : vous avez fait une faute en tapant le nom de l'environnement dans le `\begin{…}`, ou bien vous avez oublié de définir l'environnement en question.

Notes

1. J.-C. Charpentier et D. Bitouzé, *LaTeX — Synthèse et cours*, éd. Pearson Education, 2006, ISBN 2-7440-7187-0
2. C. Roland, *LaTeX par la pratique*, éd. O'Reilly, 1999, ISBN 2-84177-073-7

Écrire des mathématiques

Une des grandes forces de LaTeX est le rendu des équations mathématiques.

Les extensions `amsmath` et `amssymb` sont très utilisées, vous pouvez les inclure systématiquement.

> **Remarque :** si vous utilisez un codage unicode pour le fichier (comme `utf8`) : l'utilisation des caractères Unicode comme ×, ÷, ∀, ∃, ∈, … est actuellement déconseillée en mode mathématiques : les caractères ont des propriétés différentes (espacement, alignement vertical) selon que ce sont des quantificateurs, des opérateurs, … et LaTeX considère — pour l'instant — ces caractères Unicode comme de simples caractères. Ce point sera peut-être réglé dans le futur, mais pour l'instant, il faut utiliser les commandes à la place (`\times`, `\div`, `\forall`, `\exists`, `\in`, …).

Formules en ligne et formules centrées

On distingue trois cas :

- les formules dites « en ligne » : les symboles mathématiques sont mêlés au texte ; une telle formule commence par un signe dollar `$` et se termine par un dollar (ou commence par `\(` et finit par `\)`) ;
- les formules « centrées » : elles sont détachées du reste du texte ; une telle formule commence par `\[` et se termine par `\]` ;
- les formules centrées numérotées : comme précédemment, mais LaTeX applique une numérotation automatique. On utilise pour cela l'environnement `equation`, et l'on peut y placer une étiquette (`\label`) pour y faire référence (avec `\ref` et `\pageref`).

Par exemple

```
La fonction $f$ est définie par
\[
   f(x) = x-1
\].
On a alors
\begin{equation}
   f(x) = 0 \iff x = 1
\end{equation}
```

donne

La fonction f est définie par

$$f(x) = x - 1.$$

On a alors

$$f(x) = 0 \iff x = 1. \qquad (1.1)$$

> **Remarque :** `$…$` correspond en fait à l'environnement `math`, et `\[…\]` à l'environnement `displaymath`

Fonctions

On remarque que LaTeX utilise par défaut de l'italique. C'est en effet la forme recommandée pour les variables. Par contre, pour les fonctions on utilise du romain.

Pour cette raison, les fonctions les plus courantes disposent de leur propre instruction qui provoque leur affichage en romain, par exemple :

- les fonctions trigonométriques et hyperboliques
  - `\sin`, `\cos`, `\tan`, `\cot`
  - `\arcsin`, `\arccos`, `\arctan`, `\coth`,
  - `\sinh`, `\cosh`, `\tanh` ;
- `\ln`, `\log`, `\exp` ;
- `\max`, `\min`, `\sup`, `\inf`, `\lim` ;
- `\ker`, `\deg` ;
- `\mod`, `\bmod` (sans tabulation), `\pmod` (avec parenthèses), `\pod` (sans "mod" mais avec parenthèses).[1]

Si vous voulez utiliser une fonction non définie, il faut utiliser l'extension `amsmath` et déclarer la fonction dans le préambule (avant `\begin{document}`), avec `\DeclareMathOperator` :

```
\usepackage{amsmath}
\DeclareMathOperator{\commande}{texte}
```

où `\commande` est le nom de la commande et *texte* est le texte qui sera affiché en romain, par exemple, si l'on veut utiliser les notations obsolètes

```
\DeclareMathOperator{\asin}{asin}
```

Si l'on veut définir à nouveau une commande prédéfinie, on peut utiliser `\renewcommand{\commande}{\operatorname{texte}}`, par exemple pour remplacer le « arcsin » anglo-saxon par le « Arc sin » français on écrit :

```
\renewcommand{\arcsin}{\operatorname{Arc~sin}}
```

Polices

Si l'on veut mettre du texte normal au sein de l'équation, il faut utiliser la fonction `\text{texte}` de l'extension `amsmath` ; on peut alors utiliser toutes les fonctionnalités de LaTeX, en particulier les accents et les espaces (sinon, toutes les lettres sont collées).

Si l'on veut mettre une lettre ou quelques lettres en romain, on utilise `\mathrm{texte}`. De manière générale, les variables sont en italiques et les constantes sont en romain, en particulier :

- le nombre de Néper e, l'imaginaire i,
- les constantes de physique : la charge de l'électron e, la constante de gravité G, la célérité de la lumière dans le vide c, la constante de Planck h, la constante des gaz parfaits R, …

Les variables minuscules sont toujours en italiques, mais en typographie française, on a tendance à noter les variables capitales en romain (x mais X) ; Certaines polices ont une option permettant d'éviter d'avoir recours à `\mathrm` pour cela :

- option `upright` de l'extension `fourier` (mettre `\usepackage[upright]{fourier}` dans le préambule) ;
- option `frenchstyle` de `kpfonts` ;
- option `uppercase=upright` de `mathdesign` ;
- option `frenchmath` de `MinionPro` (police Minion Pro d'Adobe, payant).

Voir aussi l'extension `tdsfrmath` d'août 2008 (donc non incluse dans les distributions de 2008) sur le CTAN.

Pour le gras romain, on utilise `\mathbf{texte}` (pour les vecteurs en notation anglo-saxonne et les ensembles si l'on n'utilise pas la notation ajourée), `\mathsf{texte}` pour une police sans empattement, `\mathtt{texte}` pour une police à chasse fixe.

Pour les noms d'ensembles en lettres ajourées, on utilise la fonction `\mathbb`[2] de l'extension `amsfonts`, par exemple `$\mathbb{N}$` pour $\mathbb{N}$.

N.B.
auparavant, cette possibilité était fournie par l'extension `amssymb`.

On peut utiliser `\mathcal{texte}` pour les lettres « calligraphiques », par exemple `$\mathcal{LF}$` pour $\mathcal{LF}$. Si l'on charge l'extension `mathrsfs`, on a alors accès à une police cursive avec `\mathscr` *(script)*.

La commande `\mathit{texte}` met le texte en italique. Par défaut, les lettres sont déjà en italiques, mais la commande modifie l'espacement. En effet, si l'on écrit

```
$bonjour$
```

LaTeX considèrera que l'on multiplie 7 variables ($b \times o \times n \times j \times o \times u \times r$), alors que si l'on écrit

```
$\mathit{bonjour}$
```

LaTeX considère que l'on a une seule variable de sept lettres.

Lettres grecques

Pour utiliser les lettres grecques, il suffit de taper leur transcription en anglais précédée d'une contre-oblique. Par exemple :

- `\alpha` donne α ;
- `\chi` donne χ ;
- `\omega`, `\Omega` donnent ω, Ω.

Les lettres identiques aux lettres latines ne sont pas définies (le alpha capitale est identique au A, le khi capitale est identique au X). Certaines lettres ont des variantes :

- `\epsilon` donne ϵ, `\varepsilon` donne ε ;
- `\theta` donne θ, `\vartheta` donne ϑ ;
- `\pi` donne π, `\varpi` donne ϖ ;
- `\rho` donne ρ, `\varrho` donne ϱ ;
- `\sigma` donne σ, `\varsigma` donne ς ;
- `\phi` donne ϕ, `\varphi` donne φ.

Exposant, indice

Pour mettre du texte en exposant, on place le texte dans un bloc et on le fait précéder d'un chapeau « ^ ».

Pour mettre du texte en indice, on place le texte dans un bloc et on le fait précéder d'un tiret de soulignement « _ ».

par exemple :

`$ u_n = 2^n $` donne $u_n = 2^n$

`$ u_{n+1} = 2^{n+1} $` donne $u_{n+1} = 2^{n+1}$

On peut placer un objet au-dessus ou en dessous d'un autre.

Placement vertical

Instruction	Résultat
`\overset{a}{X}`	$\overset{a}{X}$
`\underset{b}{X}`	$\underset{b}{X}$
`\overset{a}{\underset{b}{X}}`	$\overset{a}{\underset{b}{X}}$

Espaces

Le mode mathématique de LaTeX ne prend pas en compte les espaces. Si l'on veut les introduire, il faut les indiquer explicitement.

Espaces en mathématiques

Instruction	Espace	Remarque
`\␣`	espace normale (justifiante)	
`\,`	espace fine	
`\:`	espace moyenne	
`\;`	grande espace	
`\!`	petite espace négative	permet le rapprochement d'objets

Fractions et racines

Les fractions s'obtiennent avec

```
\frac{numérateur}{dénominateur}
```

La racine carrée s'obtient avec

```
\sqrt{radicande}
```

et la racine énième avec

```
\sqrt[n]{radicande}
```

On peut bien sûr utiliser

```
(radicande)^{1/n}
```

Exemple :

```
\[ x_{1,2} = \frac{- b \pm \sqrt{\Delta}}{2a} \]
```

donne

$$x_{1,2} = \frac{-b \pm \sqrt{\Delta}}{2a}$$

```
$ \sqrt[4]{x^4} = |x| $
```
donne

$$\sqrt[4]{x^4} = |x|$$

Accents

Les accents classiques (diacritiques du français, de l'espagnol) s'obtiennent avec les commandes suivantes.

Accents classiques

Instruction	Résultat
`\acute{a}`	$\acute{a}$
`a', a^\prime`	$a', a^\prime$
`\grave{a}`	$\grave{a}$
`\hat{a}`	$\hat{a}$
`\widehat{ABC}`	$\widehat{ABC}$
`\tilde{a}`	$\tilde{a}$

Il existe des accents spécifiques aux mathématiques et à la physique.

Accents mathématiques

Instruction	Résultat
`\vec{a}`	$\vec{a}$
`\overrightarrow{AB}`	$\overrightarrow{AB}$
`\dot{a}, \ddot{a}`	$\dot{a}, \ddot{a}$
`\bar{a}`	$\bar{a}$
`\overline{AB}`	$\overline{AB}$
`\underline{AB}`	$\underline{AB}$
`\overbrace{1,\ldots,n}`	$\overbrace{1,\ldots,n}$
`\underbrace{1,\ldots,n}`	$\underbrace{1,\ldots,n}$

Les instruction `\imath` et `\jmath` donnent un *i* et un *j* sans point, ce qui évite la superposition d'un accent au point, par exemple `$ (O, \vec{\imath}, \vec{\jmath}) $` donne $(O, \vec{\imath}, \vec{\jmath})$.

Symboles

Pour les opérations arithmétiques, on peut utiliser `+`, `-` et `/`. Nous avons vu ci-dessus comment écrire une fraction ; on peut aussi utiliser le symbole `\div` pour avoir « ÷ ».

Pour la multiplication, on peut utiliser `\times` pour « × » et `\cdot` pour « · » ; il s'agit d'un point centré, différent du point sur la ligne « . » utilisé comme séparateur décimal par les anglo-saxons.

Opérateurs arithmétiques

Instruction	Symbole
`+`	$+$
`-`	$-$
`\times`	$\times$
`\cdot`	$\cdot$
`/`	$/$
`\frac{a}{b}`	$\frac{a}{b}$
`\div`	$\div$
`\nmid`	$a \nmid b$

:	:
\Im, \Re	$\Im$, $\Re$
\otimes	$\otimes$
\%	$\%$

Voici d'autres symboles.

Dans de nombreux cas, on peut avoir la négation d'un opérateur en utilisant `\not\opérateur`. Dans certains cas, l'opérateur nié existe, cette solution est alors préférable, par exemple

```
$ \in $ → ∈
$ \not\in $ → ∉
$ \notin $ → ∉
```

Relations

Instruction	Symbole
\equiv	$\equiv$
\approx	$\approx$
\simeq	$\simeq$
\propto	$\propto$
\leq, \geq	$\leq$, $\geq$
\leqslant, \geqslant	$\leqslant$, $\geqslant$
\ll, \gg	$\ll$, $\gg$
\lll, \ggg	$\lll$, $\ggg$
\pm	$\pm$
\mp	$\mp$
\Rightarrow, \Longrightarrow	$\Rightarrow$, $\Longrightarrow$
\Leftrightarrow, \Longleftrightarrow	$\Leftrightarrow$, $\Longleftrightarrow$

Ensembles

Instruction	Symbole
\varnothing	$\varnothing$
\cap	$\cap$
\cup	$\cup$
\subset	$\subset$
\subseteq	$\subseteq$
\in	$\in$

Quantificateurs

Instruction	Symbole
\forall	$\forall$
\exists	$\exists$

Opérateurs logiques

Instruction	Symbole
\lnot	$\lnot$
\land	$\land$
\lor	$\lor$

Géométrie

Instruction	Symbole
\wedge	$\wedge$
\angle	$\angle$
\measuredangle	$\measuredangle$
\sphericalangle	$\sphericalangle$
\perp	$\perp$

Dérivation

Instruction	Symbole
\nabla	∇
\partial	∂

Flèches

Intruction	Symbole	Remarques
\to, \mapsto	$\to, \mapsto$	
\leftarrow, \longleftarrow	$\leftarrow, \longleftarrow$	
\rightarrow, \longrightarrow	$\rightarrow, \longrightarrow$	
\leftrightarrow, \longleftrightarrow	$\leftrightarrow, \longleftrightarrow$	
\xrightarrow[a]{b}, \xleftarrow[a]{b}	$\xrightarrow[a]{b}, \xleftarrow[a]{b}$	nécessite les extensions amsmath et amssymb; le paramètre *a* est facultatif
\overrightarrow{AB}	$\overrightarrow{AB}$	
\leftrightarrows, \leftrightharpoons	$\leftrightarrows, \leftrightharpoons$	
\uparrow, \downarrow, \updownarrow	$\uparrow, \downarrow, \updownarrow$	
\nwarrow, \nearrow, \searrow, \swarrow	$\nwarrow, \nearrow, \searrow, \swarrow$	

On fera attention à la présence éventuelle d'un *s*, ainsi qu'à l'ordre des mots *left* et *right*. Si l'on met une capitale à l'instruction de la flèche, on obtient en général une flèche double.

Divers

Intruction	Symbole
\infty	∞
\hbar	$\hbar$
\circ	$\circ$
\bull	$\bullet$
\cdots, \ldots	$\cdots, \ldots$

Exemple

Définition de la limite de la fonction f, définie sur $\mathbb{R}$, en 0 :

```
\newcommand{\reels}{\mathbb{R}}
$ \lim_{x \to 0} f(x)= a \Leftrightarrow
\forall \varepsilon \in \reels^{*}_{+},\
\exist \eta \in \reels^{*}_{+} /\
\forall x \in \reels,\ |x| < \eta \Rightarrow | f(x) - a | < \varepsilon $
```

donne

$$\lim_{x \to 0} f(x)= a \Leftrightarrow \forall \varepsilon \in \mathbb{R}^{*}_{+},\ \exist \eta \in \mathbb{R}^{*}_{+} /\ \forall x \in \mathbb{R},\ |x| < \eta \Rightarrow | f(x) - a | < \varepsilon$$

Définition de la continuité de la fonction f sur $\mathbb{R}$:

```
\newcommand{\reels}{\mathbb{R}}
$ \forall a \in \reels\ \forall \varepsilon \in \reels^{*}_{+},\
\exists \eta \in \reels^{*}_{+} /\
\forall x \in \reels,\ |x-a| < \eta \Rightarrow | f(x) - f(a) | < \varepsilon $
```

donne

$$\forall a \in \mathbb{R}\ \forall \varepsilon \in \mathbb{R}^{*}_{+},\ \exists \eta \in \mathbb{R}^{*}_{+} /\ \forall x \in \mathbb{R},\ |x-a| < \eta \Rightarrow | f(x) - f(a) | < \varepsilon$$

Grands opérateurs et délimiteurs

Les grands opérateurs sont des opérateurs s'appliquant à un ensemble de valeurs, dont on indique, en général, les bornes. La borne inférieure est indiquée en indice et la borne supérieure en exposant.

Par exemple :

```
\sum_{i=0,\ i\neq j}^n u_{ij}
```

donne

$$\sum_{i=0,i\neq j}^{n} u_{ij}$$

Grands opérateurs

Instruction	Symbole
\sum	$\sum$
\prod	$\prod$
\int	$\int$
\oint	$\oint$
\bigcup	$\bigcup$
\bigcap	$\bigcap$

Autre exemple[3] :

```
\newcommand{\deriv}{\mathrm{d}}
\int \!\!\!\! \int f(x, y) \deriv x \deriv y
```

donne

$$\iint f(x,y)\mathrm{d}x\mathrm{d}y$$

Les intégrales doubles ou triples peuvent aussi être exprimées avec `\iint` ou `\iiint`, les intégrales sur les contours fermés ou surfaces fermées par `\oint` ou `\oiint`.

Les grands délimiteurs sont des délimiteurs dont la taille s'adapte à ce qu'ils contiennent. Pour cela, on place `\left` devant le délimiteur d'ouverture, et `\right` devant le délimiteur.

Exemple

```
\[ \left ( \frac{a}{b} \right ) \]
```

donne

$$\left(\frac{a}{b}\right)$$

Si l'on veut juste un délimiteur ouvrant, il faut terminer par `\right .`.

Exemple

```
\[ \left \{ … \right . \]
```

Cela s'applique aux délimiteurs suivants : `(…)`, `[…]`, `\{…\}`, `|…|`, `\|…\|` (le dernier donnant ‖…‖).

Matrices

L'écriture d'une matrice est similaire à celle d'un tableau (voir *Faire des tableaux*). Le plus simple est d'utiliser l'extension `amsmath`, on dispose alors des environnements[4] :

- `matrix` : matrice sans délimiteur ;
- `pmatrix` : matrice entre parenthèses (…) ;
- `vmatrix` : matrice entre barres |…| ;
- `Vmatrix` : matrice entre doubles barres ‖…‖ ;
- `bmatrix` : matrice entre crochets […] ;
- `Bmatrix` : matrice entre accolades {…}.

Une ligne se termine par une double contre-oblique `\\`, et sur une ligne, les coefficients sont séparés par une esperluette `&`.

Exemple

```
\begin{pmatrix}
   a_1 & b_1 \\
   a_2 & b_2
\end{pmatrix}
```

donne

$$\begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}$$

Pour les ellipses, on dispose des symboles suivants.

Ellipses pour matrices

Instruction	Symbole	Remarque
`\cdots`	$\cdots$	points centrés
`\ldots`	$\ldots$	points sur la ligne
`\vdots`	$\vdots$	
`\ddots`	$\ddots$	

Placement des objets

Pour placer finement les objets, on peut utiliser l'environnement `array` qui fonctionne comme les tableaux.

Exemple

```
  \left \{
  \begin{array}{r c l}
     AB  & = & 192 \\
     C   & = & 5\,896 \\
     DEF & = & 0,5
  \end{array}
  \right .
```

donne

$$\left\{ \begin{array}{r c l} AB & = & 192 \\ C & = & 5\,896 \\ DEF & = & 0,5 \end{array} \right.$$

Encadrer une équation

La commande `\fbox` ne fonctionne pas avec les environnements mathématiques. Pour encadrer une équation, il faut utiliser l'extension `amsmath`. S'il s'agit d'une équation en ligne, on peut alors écrire

```
On obtient donc $\boxed{p = 2\pi r}$.
```

Mais cette solution ne fonctionne pas avec les formules hors paragraphe. On peut bien sûr mettre une telle formule dans un tableau, mais on peut aussi avoir recours à l'extension `empheq`, par exemple

```
On obtient donc
\begin{empheq}[box=\fbox]{equation*}
   p = 2\pi r\text{.}
\end{empheq}
```

pour une formule non numérotée, ou bien

```
\begin{empheq}[box=\fbox]{equation}
   p = 2\pi r\text{.}
\end{empheq}
```

pour une formule numérotée. L'extension `empheq` fournit bien d'autres possibilités.

Notes

1. http://www.artofproblemsolving.com/LaTeX/AoPS_L_GuideCommands.php
2. « bb » pour *blackboard bold* (gras au tableau) : la notation a été inventée pour pouvoir écrire facilement les noms d'ensemble à la craie au tableau, le rendu du gras étant problématique
3. nous avons choisi de noter le « d » de dérivation en romaine (lettre droite) ; certains le notent en italique
4. sans `amsmath`, on dispose de l'environnement `array` qui crée une matrice sans délimiteur, et on utilise les grands délimiteurs, comme par exemple `\left ( … \right )`, voir la section suivante

Voir aussi

- Mathématiques

Wikipédia

- Aide:Formules TeX
- Table des symboles mathématiques

Liens externes

- (anglais) Mathmode (http://www.ctan.org/tex-archive/info/math/voss/mathmode/) [archive] de Hubert Voss ;
- (français) Constantes : romaines ou italiques ? (http://groups.google.com/group/fr.comp.text.tex/browse_thread/thread/ea66762fc637d761/) [archive], sur news:fr.comp.text.tex ;
- (français) **[pdf]** Eddie Saudrais, Le petit typographe rationnel (http://tex.loria.fr/typographie/saudrais-typo.pdf) [archive] (2000), ch. 2.2 « Écriture des mathématiques »

Écrire de la physique

La physique nécessite des éléments supplémentaires par rapport aux mathématiques. La plupart de ces éléments sont accessibles dans le mode mathématique, mais ils sont présentés ici. Certaines extensions sont spécifiques à la physique.

Nombres

L'extension `numprint` fournit la commande `\numprint` qui met en forme automatiquement les grands nombres[1] : avant et après le séparateur décimal, les chiffres sont groupés par trois et ces groupes sont séparés par une espace. Par exemple

```
\usepackage{numprint}
…
\numprint{123456,789123}
```

donne

123 456,789 123

En particulier, hors du mode mathématique, la commande `\numprint` permet de supprimer l'espace ajoutée après la virgule.

Mode mathématiques

En mode mathématiques, la commande `\mho` affiche le symbole du mho, ℧, et `\hbar` affiche ℏ.

Unités

Extension `siunitx`

Selon les mots de l'auteur de cette extension, Joseph Wright: CTAN (http://ctan.org/pkg/siunitx) [archive], je cite:

"Le paquet siunitx prend le meilleur des packages existants et ajoute de nouvelles fonctionnalités et une interface cohérente. Un certain nombre de nouvelles idées ont été intégrées, pour combler les lacunes des packages existants. Le paquet fournit également une compatibilité descendante avec `SIunits`, `sistyle`, `unitsdef` et `units`. L'objectif est d'avoir un package pour répondre à tous les besoins possibles liés à l'unité aux utilisateurs de LaTeX."

Extension `SIunits`

`SIunits` est incompatible avec certaines extensions, notamment la commande `\square` de l'extension `amssymb`, et la commande `\gray` de `\pstricks`. Pour éviter les conflits, on peut charger les options `squaren` et `Gray`, on utilise alors :

- `\square` avec `amssymb` et `\squaren` avec `SIunits` ;
- `\gray` avec `pstricks` et `\Gray` avec `SIunits`.

Ceci permet d'assurer la compatibilité avec du code sans `SIunits`. Nous considérerons cette hypothèse par la suite.

```
\usepackage[squaren,Gray]{SIunits}
```

Fonctionnement général

Pour écrire les unités du système international, on dispose de l'extension `SIunits` ; attention, ici, le « S » et le « I » doivent être en capitales. Les symboles des unités sont alors accessibles par des commandes portant leur nom anglais (par exemple `\kilogram`, `\meter`). Ces commandes sont utilisables dans le texte ou dans des formules mathématiques :

```
Il vaut mieux exprimer la vitesse
en mètres par seconde (\meter\per\second)
plutôt qu'en kilomètres par heure (\kilo\meter\per\hour).
On a~:
\[
   1 \kilo\meter\per\hour = \numprint{0,278} \meter\per\second
\]
```

On dispose également d'une commande

```
\unit{nombre}{\unité}
```

par exemple

```
\unit{10}{\meter}
```

pour avoir « 10 m ». L'avantage est que la typographie est correcte, notamment pour le préfixe µ (micro) : la commande `\mu` en mode mathématiques donne un caractère italique alors qu'avec `\micro` il est droit. De plus, cela gère automatiquement les espaces : espace insécable entre le nombre et l'unité, qui n'a pas la même taille que les espaces entre les composantes de l'unité.

Pour les grands nombres, on utilisera

```
\unit{\numprint{''nombre''}}{\''unité''}
```

On peut appeler l'extension avec des options :

```
\usepackage[''option'']{SIunits}
```

les principales options permettent de définir le séparateur dans le cas d'une grandeur produit de grandeurs fondamentales, lorsque l'on utilise `\usk` ou `.` à l'intérieur du deuxième bloc de `\unit` :

- `cdot` *(centered dot)* : les différentes unités sont séparées par un point centré ;
- `thinspace` : dans le cas d'une grandeur produit de grandeurs fondamentale, les différentes unités sont séparées par une petite espace ;
- `mediumqspace` : le nombre et l'unité sont séparés d'une espace moyenne ;
- `thickqspace` : le nombre et l'unité sont séparés d'une grande espace.

Par exemple :

```
\usepackage[squaren, Gray, cdot]{SIunits}
```

Unités de base

Les unités SI et dérivées sont obtenues simplement en écrivant leur nom anglais précédé de la contre-oblique.

Unités SI dans `SIunits`

Unité	Commande	Résultat
kilogramme	`\kilogram`	kg
mètre	`\metre`, `\meter`	m
seconde	`\second`	s
ampère	`\ampere`	A
kelvin	`\kelvin`	K
mole	`\mole`	mol
candela	`\candela`	cd

Les unités dérivées sont : `\hertz` (Hz), `\pascal` (Pa), `\newton` (N), `\joule` (J), `\watt` (W), `\coulomb` (C), `\volt` (V), `\farad` (F), `\ohm` (Ω), `\siemens` (S), `\weber` (Wb), `\tesla` (T), `\henry` (H), `\celsius` (°C), `\lumen` (lm), `\lux` (lu), `\becquerel` (Bq), `\Gray`[2] (Gy), `\sievert` (Sv).

Un certain nombre d'unités non-Si sont également définies : `\angstrom` (Å), `\arcminute` (′), `\arcsecond` (″), `\are` (a), `\atomicmass` (u), `\barn` (b), `\bbar` (b), `\bel` (B), `\curie` (Ci), `\dday` (d), `\degree` (°), `\electronvolt` (eV), `\gal` (Gal), `\gram` (g), `\hectare` (ha), `\hour` (h), `\liter` (L), `\litre` (l), `\minute` (min), `\neper` (Np), `\rad` (rad), `\rem` (rem), `\roentgen` (R), `\rperminute` (r/min), `\tonne` (t).

Préfixes multiplicatifs

Les préfixes s'utilisent de la même manière. On a donc : `\yocto` (y, 10^{-24}), `\zepto` (z, 10^{-21}), …, `\micro` (µ, 10^{-6}), `\milli` (m, 10^{-3}), `\centi` (c, 10^{-2}), `\deci` (d, 10^{-1}), `\deca` (da, 10), `\hecto` (h, 10^{2}), `\kilo` (k, 10^{3}), `\mega` (M, 10^{6}), …, `\zetta` (Z, 10^{21}), `\yotta` (Y, 10^{24}).

Si l'on intègre un « d » à la fin du préfixe, cela génère la puissance de 10, par exemple

`\unit{5}{\micro\metre}` donne 5 µm, mais
`\unit{5}{\microd\metre}` donne 5 10^{-6} m

Si l'on veut mettre un point centré, il faut écrire `\unit{5}{.\microd\metre}`, mais alors, on a un espace puis un point. On peut utiliser

```
$\unit{5 \cdot 10^{-6}}{\meter}$
```

Unités composées

L'extension contient un certain nombre d'unités composées. Il suffit d'utiliser le nom en anglais, en un seul mot, par exemple

- `\squaremeter` : m^2 ;
- `\cubicmeter` : m^3 ;
- `\squaremeterpercubicmeter` : m^2/m^3 ;
- `\kilogramsquaremeter` : $kg \cdot m^2$;
- `\kilogrampersquaremeter` : kg/m^2 ;
- `\kilogrampersquaremeternp` : $kg \cdot m^{-2}$;
- `\kilowatthour` : kWh ;
- …

On peut aussi composer ses propres unités :

- pour multiplier des unités entre elles, il suffit de les mettre les unes après les autres, elles sont alors accolées ; on peut les séparer par un point « . » ou par la commande `\usk` *(unit skip)* pour utiliser le délimiteur défini en option (si l'on a choisi l'option `mediumspace`, le point sera remplacé par une espace moyenne) ;
- pour placer une barre de fraction, on utilise `\per` ;
- pour élever à la puissance :
  - -1 : on fait précéder l'unité par `\reciprocal`, par exemple `\reciprocal\second` pour s^{-1}
  - 2 : on fait précéder l'unité par `\squaren`[3] ou suivre par `\squared`, par exemple `\squaren\meter` ou `\meter\squared` pour m^2 ;
  - 3 : on fait précéder l'unité par `\cubic` ou suivre par `\cubed`, par exemple `\cubic\meter` ou `\meter\cubed` pour m^3 ;
  - -2 : on fait précéder l'unité par `\rpsquare` ou suivre par `\rpsquared`, par exemple `\rpsquare\meter` ou `\meter\rpsquared` pour m^{-2} ;
  - -3 : on fait précéder l'unité par `\rpcubic` ou suivre par `\rpcubed`, par exemple `\rpcubic\meter` ou `\meter\rpcubed` pour m^{-3} ;
  - une puissance quelconque : `\power{`*`unité`*`}{`*`puissance`*`}`, par exemple `\power{m}{\mathit{n}}` pour m^n.

Par exemple

`\unit{25}{\kilogram.\meter \per \squaren \second}` donne 25 $kg \cdot m/s^2$
`\unit{25}{\kilogram.\meter.\rpsquare \second}` donne 25 $kg \cdot m \cdot s^{-2}$

Extension `sistyle`

L'utilisation de l'extension `sistyle` — ici, le « s » et le « i » sont en bas-de-casse — est beaucoup plus simple que `SIunits`. On utilise la commande

```
\SI{nombre}{unité}
```

mais ici, l'unité est composée comme en mode mathématique. Cela est laissé à l'entière responsabilité de l'utilisateur. Par exemple

```
\SI{1}{N}=\SI{1}{kg.m/s^2}
```

Le point est remplacé par un point centré ; si l'on veut utiliser le point comme séparateur décimal, on utilise `\pnt`, par exemple

`\SI{}{MPa^{0 \pnt 5}}` pour $(MPa)^{0.5}$.

La commande `\tfrac{`*`dividende`*`}{`*`diviseur`*`}` permet d'écrire une rapport d'unités sous la forme d'une petite fraction.

L'extension fournit des commandes pour certaines unités qui ne sont pas des lettres latines :

- `\angstrom` : Å ;
- `\micro` : µ ;
- `\ohm` : Ω ;
- `\degC` : °C ;
- `\degF` : °F ;
- `\arcdeg` : ° ;
- `\arcmin` : ' ;
- `\arcsec` : ".

La commande `\num*{`*`nombre`*`}` formate les nombres, en séparant les groupes de trois chiffres par une espace. Elle fournit également une notation simplifiée des puissances de 10, avec `e` :

`1e2` donne 1×10^2
`- e -0,5` donne $-10^{-0.5}$

Pour avoir la virgule comme séparateur décimal, il faut employer

```
\SIdecimalsign{,}
```

La commande `\num` a le même effet, mais utlise des chiffres bas-de-casse (ceux obtenus avec `\oldstylenums`).

L'extension fournit également une mise en forme des angles :

```
\ang{degrés; minutes; secondes}
```

par exemple

`\ang{3; 2; 1}` donne 3°2'1"

Réactions nucléaires

Pour pouvoir écrire une réaction nucléaire, il est utile de savoir utiliser les instructions d'indices et d'exposants. Sauf qu'il faut les placer avant l'élément concerné et pas après comme en mathématique. Pour cela, il suffit simplement d'écrire l'instruction voulue avant. Par exemple `_{6}^{12}C_{6}` nous donne ${}_{6}^{12}C_{6}$. On peut donc facilement écrire des réactions nucléaires complètes en utilisant cette méthode. Ainsi `_{7}^{14}N + _{0}^{1}n \longrightarrow _{1}^{1}p + _{6}^{14}C` permet d'écrire ${}_{7}^{14}N + {}_{0}^{1}n \longrightarrow {}_{1}^{1}p + {}_{6}^{14}C$. Il semblerait qu'il soit préférable d'écrire en premier la partie concernant l'indice et ensuite d'écrire la partie concernant l'exposant. Par exemple, on écrira `_{6}^{12}C` (ce qui nous donne ${}_{6}^{12}C$) à la place de `^{12}_{6}C` (ce qui nous donne ${}_{6}^{12}C$) ; le rendu est plus parlant.

Quand l'indice et l'exposant n'ont pas le même nombre de caractères --- 1 pour "6" et 2 pour "12" --- on peut les aligner à droite à l'aide de la commande `\phantom`, comme dans `_{\phantom{1}6}^{12}C`, qui donnera un résultat plus joli que ${}_{6}^{12}C$, sans espace sous le "2".

Il est possible de définir la commande \newcommand{\noyau}[3]{\prescript{#2}{#3}{\mathrm{#1}}} dans l'entête. Le paquet **mathtools** définit la commande \prescript. On doit donc aussi l'inclure pour pouvoir utiliser cette commande. L'avantage de cette commande est que, par exemple pour le carbone 12, le 6 sera correctement aligné à droite, ce qui n'est pas le cas avec les commandes du paragraphe précédent. Pour écrire un noyau de carbone 12, vous devez simplement taper `\noyau{C}{12}{6}` dans un environnement mathématique. Il est possible de laisser l'un des paramètres libre en écrivant par exemple `\noyau{C}{12}{}` qui afficher ${}^{12}\mathrm{C}$.

Notation de Dirac, bra et ket

Pour la physique quantique, les instructions utiles sont `\langle` pour faire $\langle$, `\rangle` pour faire $\rangle$ et le tube |.

On peut utiliser l'extension `braket`, qui fournit

- les commandes `\bra`, `\ket` et `\braket` pour des symboles de taille fixée,
- et les commandes `\Bra`, `\Ket` et `\Braket` pour des symboles extensibles selon leur contenu.

On utilisera comme suit :

```
\bra{a}
\ket{b}
\braket{a|P|b}
```

On peut aussi choisir de créer soi-même des commandes `bra` et `ket`, pour plus de souplesse. Par exemple, on peut mettre dans l'en-tête :

```
\usepackage{xspace}

\newcommand{\ket}[1]{\ensuremath{|#1\rangle}\xspace}
\newcommand{\bra}[1]{\ensuremath{\langle #1|}\xspace}
```

Ainsi, `\ket{n_i}` donne $|n_i\rangle$, `\bra{p}` donne $\langle p|$ et `\bra{u_i} \hat{A} \ket{u_j}` donne $\langle u_i|\hat{A}|u_j\rangle$.

On peut ensuite créer de nouvelles commandes utilisant `\bra` et `\ket`, par exemple

```
\newcommand{\uiacuj}{\ensuremath{\bra{u_i}\hat{A}^\dagger \ket{u_j}}\xspace}
```

et donc `\uiacuj` donne $\langle u_i|\hat{A}^\dagger|u_j\rangle$. Ou encore, pour rendre cette dernière commande paramétrable :

```
\newcommand{\elemm}[3]{\ensuremath{\bra{#1}\hat{#2}\ket{#3}}\xspace}
```

par exemple, `\elemm{a}{W}{b}` donne $\langle a|\hat{W}|b\rangle$, et `\elemm{a_n^{(1)}}{H_0}{b_p^{(2)}}` donne $\langle a_n^{(1)}|\hat{H}_0|b_p^{(2)}\rangle$.

Pour écrire un produit scalaire, de la même façon :

```
\newcommand{\psh}[2]{\ensuremath{\langle #1|#2\rangle}\xspace}
```

et donc `\psh{a}{b}` donne $\langle a|b\rangle$. Ou encore `\psh{x-\ell}{\varphi}` qui donne $\langle x-\ell|\varphi\rangle$.

Un joli exemple, pour le plaisir .En plus des commandes ci-dessus, on peut faire du zèle et créer les commandes suivantes

```
\newcommand{\upp}[1]{\ensuremath{^{(#1)}}\xspace}
\newcommand{\sqmod}[1]{\ensuremath{|#1|^2}\xspace}
```

Et alors

```
\ket{n} \upp{1} \sim
\ket{n} +
\sum_{p\neq n} \frac{\sqmod{\elemm{p}{W}{n}}}{E_n\upp{0}-E_p\upp{0}}
\ket{p}
```

donne

$$|n\rangle^{(1)} \sim |n\rangle + \sum_{p\neq n} \frac{\langle p|\hat{W}|n\rangle}{E_n^{(0)} - E_p^{(0)}}|p\rangle$$

ce qui, sans les macros, s'écrirait in extenso

```
| n \rangle^{(1)} \sim
| n \rangle +
\sum_{p\neq n}\frac{\langle p | \hat{W} |n \rangle}{E_n^{(0)}-E_p^{(0)}}
| p \rangle
```

Notes

1. dans la documentation Babel du 18 mars 2010, Daniel Flipo recommande l'utilisation de la commande `\numprint` au lieu de la commande `\nombre`, voir [[3] (http://daniel.flipo.free.fr/frenchb/frenchb2-doc.pdf)] en bas de page 4.
2. avec l'option `Gray` lors de l'appel de l'extension
3. avec l'option `squaren` lors de l'appel de l'extension

Écrire des formules chimiques

Il existe plusieurs extensions dédiées à l'écriture de la chimie. On peut en trouver plusieurs dans le TeXCatalog (http://texcatalogue.sarovar.org/bytopic.html#chem) [archive].

Note

Le dessin des molécules chimiques est décrit au chapitre *Dessiner des molécules*.

Avec `chemist`

L'extension `chemist` fournit des commandes et environnements s'utilisant comme les modes mathématiques, mais affichant les lettres romaines (droites) et non italiques. L'appel de l'extension se fait par

```
\usepackage{chemist}
```

Pour une formule dans le texte, on utilise la commande `\chemform{`*`formule`*`}`, par exemple

```
\chemform{Fe_2O_3}
```

donne

Fe_2O_3

L'environnement `chemmath` est l'équivalent de `\[…\]` (formules centrées), et l'environnement `chemeqn` est l'équivalent de l'environnement mathématique `equation` (formules centrées numérotées) :

```
\begin{chemmath}
.
.
.
\end{chemmath}
\begin{chemeqn}
.
.
.
\end{chemeqn}
```

Pour les équations, on utilise les flèches habituelles : `\rightarrow`, `\longrightarrow`, `\rightleftarrows`, `\rightleftharpoons`, … (voir ci-après). Par exemple,

```
\begin{chemmath}
CH_4 + 2O_2 \longrightarrow CO_2 + 2H_2 O
\end{chemmath}
```

donne

$$CH_4 + 2O_2 \longrightarrow CO_2 + 2H_2O$$

Notons que cette extension, au moins à ce jour (v2.00a du 3 novembre 2000), modifie le placement des flèches de vecteur dans le mode mathématique, ce qui pose problème avec les capitales (par exemple `$\vec{F}$`).

Avec `mhchem`

L'extension `mhchem` permet d'écrire simplement des équations de réaction[1], mais pas de dessiner des molécules. L'appel de l'extension au complet se fait par

```
\usepackage[version=3]{mhchem}
```

L'option `version=3` est importante pour bénéficier de toutes les fonctionnalités.

La commande principale est `\ce{`*`formule`*`}`. Notons que :

- les + et - sont placés automatiquement en exposant
  `\ce{Fe^{2}+}` → $\ce{Fe^{2}+}$; on peut aussi utiliser la notation habituelle `^{…}` ;
- les coefficients après des symboles chimiques ou les parenthèses sont placés directement en indice, les coefficients avant sont placés sur la ligne

`\ce{2Sb2O3}` → $2Sb_2O_3$; on peut aussi utiliser la notation habituelle `_{…}` ;
- l'écriture des fractions est simple
  `\ce{1/2H2O}` → $\frac{1}{2}H_2O$;
- la notation mathématique en indice et exposant est utilisable, y compris avant un symbole chimique, comme pour les isotopes
  `\ce{^{227}_{90}Th+}` → $^{227}_{90}Th^+$;
- la liaison simple se marque avec un `-` ou `\sbond`, la double avec un `=` ou `\dbond`, et la triple avec un `#` ou `\tbond` ;
- la flèche de réaction se marque `->` ;
  - on peut mettre du texte au dessus avec `->[`*`texte au dessus`*`]`,
  - du texte au dessus et en dessous avec `->[`*`texte au dessus`*`][`*`texte en dessous`*`]` ;
- la double flèche d'équilibre se note `<=>` ;
- un chapeau `^` entouré de deux espace indique un dégagement de gaz ↑ ; un `v` entouré de deux espaces indique une précipitation ↓ ;
- un astérisque « * » ou un point « . » est transformé en un point centré « · ».

Par contre, l'extension ne gère pas très bien les formules contenant des coefficients sous forme de lettre, par exemple mC_nH_{2n+2}. Mais on peut mêler les `\ce{…}` au sein d'expressions mathématiques, ou des expressions mathématiques au sein d'un `\ce{…}`, pour contourner le problème.

Si l'on veut mettre une formule à l'intérieur d'une formule, il vaut mieux utiliser `\cf{…}` à l'intérieur du `\ce{…}`. La commande `\cf{…}` est quasiment équivalente à `\ce{…}`.

Par exemple

```
\ce{CH4 + 2O2 -> CO2 + 2H2O}

$\ce{C}_n \ce{H}_{2n+2} + \frac{3n+1}{2}\ce{O2 ->[\Delta G]} n \ce{CO2} + (n+1)\ce{ H2O}$

\ce{Ag+ + Cl- -> AgCl v}
```

donnent

$$CH_4 + 2O_2 \longrightarrow CO_2 + 2H_2O$$
$$C_nH_{2n+2} + \frac{3n+1}{2}O_2 \longrightarrow \Delta G n CO_2 + (n+1)H_2O$$
$$Ag^+ + Cl^- \longrightarrow AgCl\downarrow$$

Avec ppchtex

> **Remarque :** l'extension livrée avec MacTeX 2007 est erronée ; il faut télécharger la version à jour (ftp://cam.ctan.org/tex-archive/macros/context/current/cont-ppc.zip) [archive] et l'installer dans `~Library/texmf`

L'extension *ppchtex* se compose de deux extensions : `m-pictex` et `m-ch-en`. On place donc dans le préambule

```
\usepackage{etex}
\usepackage{m-pictex,m-ch-en}
```

(L'extension `etex` permet de rajouter des « registres », ce qui peut être nécessaire avec PPCHTeX.)

Une formule en ligne s'écrit sous la forme

```
\chemical{élément 1, élément 2, …, élément n}
```

où les éléments peuvent être :

- un atome ou un groupement d'atomes : on utilise la notation chimique classique (`C` pour le carbone, `CH_4` pour le méthane) ;
- une liaison simple : `-` ou `SINGLE` ;
- une liaison double : `--` ou `DOUBLE` ;
- une liaison triple : `---` ou `TRIPLE` ;
- une espace fine : `\,` ;
- une flèche de réaction : `->` ou `GIVES` ;
- deux flèches tête-bêche d'équilibre ⇄ : `<->` ou `EQUILIBRIUM`
- une double-flèche ↔ : `<>` ou `MESOMERIC` ;
- un plus : `+` ou `PLUS`.

Lorsque la formule est dans un environnement mathématique hors paragraphe (`\[…\]` ou environnement `equation`), on peut placer du texte en dessous des éléments :

```
\chemical{texte principal}{texte dessous}
\chemical{texte principal}{texte dessus}{}
\chemical{texte principal}{texte dessus}{texte dessous}
```

par exemple

```
\[
   \chemical{2H_2}{dihydrog\grave{e}ne} \chemical{+} \chemical{O_2}{dioxyg\grave{e}ne}
   \chemical{->}{combustion}{}
   \chemical{2H_2O}{eau}
\]
```

pour obtenir

$$\underset{\text{dihydrogène}}{2H_2} + \underset{\text{dioxygène}}{O_2} \xrightarrow{\text{combustion}} \underset{\text{eau}}{2H_2O}$$

Si la formule est en texte, on peut placer du texte au dessus de flèches :

```
\startchemical
   \chemical[''flèche''][''texte dessus'']
\endchemical
```

ou *flèche* est `GIVES`, `MESOMERIC` ou `EQUILIBRIUM` (on ne peut pas utilise `->`, `<>` ou `<->`). La flèche est alors considérée comme faisant partie d'une molécule, ce qui donne de grandes espaces avant et après ; par contre, la flèche est plus longue.

```
\chemical{2H_2} \chemical{PLUS} \chemical{O_2}
\startchemical
   \chemical[GIVES][combustion]
\stopchemical
\chemical{2H_2O}
```

Avec le mode mathématiques

Pour écrire des formules chimiques, on peut bien sûr utiliser les mathématiques (voir *LaTeX/Écrire des mathématiques*), en retenant les points suivants :

- on utilise `\mathrm{…}` pour avoir une écriture en romain ; comme cela concerne tous les symboles chimiques, cela devient vite fastidieux ;
- pour les formules semi-développées en ligne, on peut utiliser pour `-` les liaisons simples, `=` pour les liaisons doubles et `\equiv` pour les liaisons triples ; pour les dessins de molécules, on envisagera d'utiliser un logiciel de dessin et d'intégrer l'image ainsi produite ;
- les flèches de réaction sont obtenues par `\rightarrow`, `\longrightarrow` et `\rightleftarrows` (ou `\rightleftharpoons`) ;
- avec l'extension `amsmath`, on peut utiliser la flèche `\xrightarrow{`*texte dessus*`}`, `\xrightarrow[`*texte dessous*`]{}` et `\xrightarrow[`*texte dessous*`]{`*texte dessus*`}` ;
- penser aux instructions de placement `\overset{`*texte dessus*`}{`*texte principal*`}` et `\underset{`*texte dessous*`}{`*texte principal*`}`.

Exemples :

```
\newcommand{\chimie}[1]{$\mathrm{#1}$}
\newcommand{\chimiecite}[1]{\[\mathrm{#1}\]}
\newcommand{\nchim}{\mathit{n}}

\chimiecite{CH_4 + 2O_2 \longrightarrow CO_2 + 2H_2 O}

\chimiecite{ C_{\nchim} H_{2\nchim+2} + \frac{3\nchim + 1}{2}O_2 \xrightarrow{\Delta G}
   \nchim CO_2 + (\nchim+1)H_2 O} \]

\chimiecite{Ag^+ + Cl^- \rightarrow \underset{\searrow}{AgCl}}
```

donnent

$$CH_4 + 2O_2 \longrightarrow CO_2 + 2H_2O$$

$$C_nH_{2n+2} + \frac{3n+1}{2}O_2 \xrightarrow{\Delta G} nCO_2 + (n+1)H_2O$$

$$Ag^+ + Cl^- \rightarrow \underset{\searrow}{AgCl}$$

Pour les équations dans le texte, on peut mixer le texte normal et les mathématiques, mais on se prive alors de la possibilité d'utiliser les formules centrées, numérotées ou non (`\[…\]` et environnement `equation`). Le deuxième exemple devient alors

```
C$_n$H$_{2n+2} + \frac{3n + 1}{2}$O$_2 \xrightarrow{\enthreac} n$CO$_2 + (n+1)$H$_2$O
```

Avec `chemtex`

L'extension ChemTeX se trouve à l'adresse suivante :

> http://www.ctan.org/tex-archive/macros/latex209/contrib/chemtex/

Les formules chimiques se tapent simplement en mode mathématique comme évoqué ci-dessus.

ChemTeX consiste en des macros de mise en forme. Par exemple, pour tracer une flèche de réaction :

```
\cto{texte au dessus}{texte en dessous}{longueur}
```

où *longueur* est le nombre de caractères du texte le plus grand (indices et exposants compris). Par exemple :

```
\[
  HC \equiv CH + H_{2}O
  \cto{Hg^{++}}{18\%\ H_{2}SO_{4},\ 90^\circ}{14}
  CH_{3}-CHO
\]
```

pour

$$HC \equiv CH + H_2O \xrightarrow[18\%\ H_2SO_4,\ 90^\circ]{Hg^{++}} CH_3 - CHO$$

Notes

1. ainsi que les phrases d'avertissement sur les risques chimiques

Voir aussi

Wikilivres

- LaTeX/Dessiner_avec_LaTeX/Dessiner_des_molécules

Liens externes

- Pour dessiner un tableau de classification périodique : Mendeleiev, il pue ? (http://groups.google.com/group/fr.comp.text.tex/browse_thread/thread/a18c8ea09ff803bf/81bb8dde609b5352) [archive], discussion du 26 août sur news:fr.comp.text.tex

Créer une feuille de style

Nous avons laissé jusqu'ici LaTeX gérer toute la mise en forme. Il est possible de modifier la manière de faire de LaTeX, tout en gardant le côté « automatique » ; la démarche est similaire à la notion de feuille de style en HTML/CSS. Nous allons traiter ceci sous la forme d'un exemple.

Consignes éditoriales

Nous avons choisi, ou bien il nous a été imposé, les choses suivantes :

- page au format A4 ;
- fonte Times ;
- interligne simple ;
- le titre est en corps 16 pt, le sous-titre en corps 14, le texte en corps 12, les notes en corps 10 ;
- le titre, centré, est détaché du texte par trois sauts de paragraphe ;
- les paragraphe sont en alinéa avec un retrait de 1,25 cm ;
- les intertitres (un seul niveau) sont en italiques, détachés par un saut de paragraphe avant et après, alignés à gauche.

Il nous faut d'abord « traduire » certaines consignes en termes typographiques pour retrouver les commandes et extensions correspondantes dans LaTeX :

- « saut de paragraphe » : espace vertical de un quadratin (`1 em`) ;
- « intertitres (un seul niveau) » : titre de section.

Réalisation du préambule

Comme le document ne peut comporter qu'un seul niveau de titre, on peut choisir la classe `article` ; on indique le corps du texte :

```
\documentclass[a4paper, 12pt]{scrartcl}
```

Notons que l'on a imposé un corps de 12 pt au texte « normal » ; pour l'instant, c'est LaTeX qui fixe les corps des autres parties (titres, notes), de manière proportionnelle. Avec le paramètre `12pt`, le corps des notes est de 10 pt, il n'y a donc rien pour ce cas-là. Par contre, le titre de l'article est dans le meme corps que la commande `\Huge`, c'est-à-dire ici 25pt : la commande `\large`, quant à elle, donne un corps de 14 pt et peut donc être utilisé pour le sous-titre.

Les marges sont définies simplement par l'extension `geometry`, et la fonte par l'extension `times` :

```
\usepackage[margin=2.5cm]{geometry}
\usepackage{times}
```

La définition du titre de document peut être ajustée avec l'extension `titling`. Pour redéfinir les corps, nous utilisons la commande `\fontesize{`*`corps1`*`}{`*`corps2`*`}\selectfont`, où *corps1* est le corps du texte et *corps2* est le corps utilisé pour les interlignes.

```
\usepackage{titling}

\pretitle{\begin{center}\fontesize{16pt}{16pt}\selectfont}
\posttitle{\par\end{center}\vskip 3em}
```

La classe `article` ne dispose pas de commande pour générer un sous-titre. On aurait pu utiliser la classe `scrartcl` à la place, mais l'extension `titling` ne permet pas de redéfinir le sous-titrage, on créera donc le sous-titre à la main. La création à la main est déconseillée, car cela ne permet pas de garantir l'uniformité entre les documents ; toutefois, ici, cela n'intervient qu'une seule fois par document, et automatiser l'opération nous entraînerait trop loin.

Notes

Voir aussi

À l'aide !

Si LaTeX est un programme très stable (il ne plante pas), il y a en revanche des possibilités de faire des erreurs en entrant les instructions. Cela va se manifester par des messages d'erreur et un résultat non conforme à vos désirs, voire pas de résultat du tout…

Voir aussi Premier exemple > Déroulement de la compilation.

Comme il s'agit d'un logiciel libre, vous avez une communauté du Web qui sera prête à vous aider, à condition que vous vous montriez poli !

Nous allons voir quelques erreurs communes, comment les éviter, et comment demander de l'aide sur le Web.

Utiliser un éditeur de texte adapté

Nous avons vu précédemment que le choix d'un éditeur de texte adapté était fondamental (voir *Installer LaTeX > Choisir un éditeur de texte*) :

- il permet de repérer les parenthèses, crochets ou environnements que l'on aurait ouvert et pas refermé ;
- il permet d'insérer les instructions les plus courantes, réduisant le risque de faute de frappe ;
- la coloration syntaxique permet de mieux repérer les instructions et appariements d'accolades et crochets, donc de déceler les sources d'erreur ;
- en facilitant la mise en forme du code (indentations), il le rend plus lisible et donc rend plus facile son déverminage (voir *Éléments de base > Structuration du code source*).

Faire des essais

Vous voudrez probablement, à un moment ou à un autre, faire des essais, pour tester une commande que vous venez de découvrir ou de créer par exemple. Deux outils peuvent vous être utiles : la classe de document `minimal` et l'extension `lipsum`.

La classe de document `minimal`, comme son nom l'indique, contient le minimum requis pour faire un texte, et rien d'autre. De ce fait, tout ce que vous utilisez dans votre document ne marche pas, mais l'utilisation d'une classe minimale permet de faire fonctionner ce qui est « universel ». Par exemple, vous ne pouvez pas mettre de sectionnement (`\chapter{…}`, `\section{…}`, …), les listes s'affichent sans symbole ou numéro en tête d'alinéa, les tableaux n'ont pas de filet, …

L'extension `lipsum` fournit la commande `\lipsum`, qui génère sept paragraphes d'un texte en pseudo-latin (le fameux *« Lorem ipsum… »*[1]), occupant environ une page. Le texte complet comporte 150 paragraphes, et s'obtient avec `\lipsum[1-150]`, ce qui représente une vingtaine de pages. Vous pouvez ne mettre que le premier paragraphe en utilisant `\lipsum[1]`. Ceci permet de faire du remplissage.

Voici par exemple un document de test minimal :

```
\documentclass{minimal}

\usepackage{lipsum}

\begin{document}
   \lipsum
\end{document}
```

Problèmes de rendu

Les fichiers générés, utilisés pour la visualisation ou l'impression, sont au format *device independent* (fichier `.dvi`), Adobe PostScript (fichier `.ps`) ou Adobe Portable Document Format (fichier `.pdf`). Ce ne sont pas des « images figées » qui sont simplement affichées par le visualiseur ; le visualiseur « redessine » à chaque fois le document à partir des informations contenues dans le fichier.

De fait, les différents programmes de visualisation n'ont pas *exactement* le même comportement, et l'on peut avoir un affichage légèrement différent de ce à quoi on s'attend. L'impression est elle en général correcte (elle est gérée par le pilote d'impression, ou *driver*, qui interprète « correctement » le fichier).

Ce genre de problème ne se produit pas pour du texte simple, bien que certains visualiseur peuvent avoir un rendu assez mauvais des polices Computer Modern et Extended Computer Modern. Il va surtout survenir pour des fond colorés, où l'on peut voir des décalages entre le pavé de couleur et le filet (contour de la boîte), ou bien si l'on fait des dessins (problèmes d'épaisseur de trait).

Face à un tel problème, on peut

- imprimer la page incriminée et la comparer au rendu à l'écran ;
- essayer avec un autre visualiseur, et comparer les rendus ;
- générer un fichier de l'autre format (`.ps` si l'on utilise le `.pdf`, `.pdf` si l'on utilise le `.ps`) et comparer les rendus ;

pour voir si le problème vient du rendu ou du fichier (et donc du code que vous avez tapé ou bien d'une bogue de l'extension employée).

Nettoyage des fichiers générés

Parfois, vous pensez avoir corrigé le problème, mais un message d'erreur persiste. Les fichiers générés (`.aux`, éventuellement `.bbl` et `.blg` pour la

bibliographie, et `.idx` et `.ind` pour un index) peuvent contenir des erreurs générées par la compilation de l'ancienne version du document LaTeX. Le fait d'effacer ces fichiers — mais en faisant attention à bien conserver le fichier `.tex` (il portent tous le même nom …) — peut résoudre le problème.

Consultation du journal des erreurs

Lors de la compilation, `latex` (ou `pdftex`) génère des messages indiquant le bon déroulement de l'opération, ou bien des avertissement et messages d'erreur.

Si vous compilez en ligne de commande, les messages s'affichent dans la fenêtre du *shell*. Il peuvent aussi s'afficher dans une fenêtre dédiée de votre éditeur de texte. Sinon, il sont placés dans un fichier portant le nom du fichier source mais l'extension `.log`, appelé « fichier de *log* » ou « journal des erreurs ».

La première chose à faire est donc de consulter le journal des erreurs afin de voir les problèmes. Il s'agit d'un fichier de texte, s'ouvrant avec n'importe quel éditeur de texte.

Démarche de déverminage

La première chose à faire est de tenter de résoudre les erreurs indiquées dans le journal des erreurs.

Voir *Chasse aux erreurs*

Parfois, vous n'arriverez pas à résoudre certaines erreurs. La première chose à faire est d'isoler le passage problématique. En effet, une erreur peut se manifester bien plus loin ; par exemple, si vous oubliez de fermer une accolade ou un environnement, tout le texte qui suit est considéré comme influencé par la commande ou faisant partie de l'environnement, et le message peut apparaître lorsque le contenu devient incompatible avec la commande ou l'environnement.

Ordre des extensions

L'ordre dans lequel les extensions sont déclarées dans le préambule peut avoir de l'importance. En effet, certaines extensions redéfinissent le comportement d'une ou plusieurs fonctions de LaTeX, une fonction peut ainsi se voir redéfinir par plusieurs extensions.

De manière générale, il faut :

1. placer les extensions `inputenc` et `fontenc` en premier ;
2. puis, placer les extensions qui modifient la mise en page du document, comme par exemple `geometry` ;
3. placer `babel` en dernier.

Il existe des exceptions, qui sont en général mentionnées dans la documentation de l'extension. Par exemple, l'extension `hyperref` doit être placée en dernier (après `babel`) dans le préambule.

Isoler le passage problématique

Le message d'erreur indique le numéro de ligne où se produit l'erreur. Si vous avez un éditeur de texte adapté, vous retrouverez facilement cette ligne. Il faut alors isoler le passage :

- créez un fichier `.tex` vierge ; on le baptise traditionnellement `ecm.tex`, comme « exemple complet et minimal » ;
- copiez le préambule du fichier en cause ;
- copiez le passage en cause.

Le passage en cause contiendra évidemment la ligne citée, mais

- si cette ligne est à l'intérieur d'un bloc, on mettra tout le bloc et éventuellement la ou les instructions qui agissent sur le bloc ; une bonne précaution consiste à garder le paragraphe en cours ;
- si cette ligne est dans un environnement, on mettra tout l'environnement.

Puis, on recompile cet ECM. Si l'erreur ne se produit pas, c'est que le problème se trouve ailleurs, on fera une recherche par dichotomie (cf. ci-après). Sinon, on va essayer de simplifier encore l'ECM afin d'isoler vraiment l'erreur. Pour cela,

1. on enlève les extensions inutiles, pour ne garder que `inputenc` (si l'on a des caractères accentués dans le source) et les extensions spécifiques au passage (commandes ou environnements non-standard) ;
2. on enlève tout le texte inutile, précédant ou suivant l'instruction à problème ; pour un tableau, on ne gardera par exemple qu'une ligne du tableau, et on mettra un texte simplifié dans les autres cases du tableau.

On enregistre et on recompile après chaque simplification : si le problème disparaît, c'est que l'on a enlevé une partie contribuant au problème ; il faut alors la remettre et continuer la simplification.

Cette démarche peut vous permettre de déceler l'erreur. Sinon, vous disposez d'un fichier ECM que vous pourrez soumettre à la communauté Internet.

Recherche par dichotomie

Si le problème ne se situe pas dans le passage pointé par le journal d'erreur, il va falloir chercher le passage incriminé… La stratégie la plus rapide est la dichotomie :

1. on renomme le fichier en question en `ecm.tex` (par exemple) ;
2. on supprime tout ce qui suit le passage problématique (en conservant le bloc et/ou l'environnement en cours, et bien sûr le `\end{document}` final) ; on enregistre et on recompile pour vérifier que l'erreur se situe bien avant ;
3. soit *n* le numéro de la ligne signalé ; on supprime les lignes entre le `\begin{document}` et la ligne *n*/2 (arrondie au dessus ou en dessous si *n* est impair), en faisant attention de ne pas couper un paragraphe ni un environnement, *a fortiori* un bloc ;
4. on enregistre et on recompile ;
   - si l'erreur disparaît, c'est qu'une partie du problème se trouve dans la partie supprimée ; on remet donc cette partie et on supprime la partie entre la ligne *n*/2 et la ligne en cours (en ne coupant ni un paragraphe, ni un environnement, ni un bloc) ; on reprend au point 3 (le numéro de ligne *n* a changé puisqu'il y a moins de lignes) ;
   - si l'erreur est toujours là, on reprend au point 3 (le numéro de ligne *n* a changé puisqu'il y a moins de lignes).

En quelques étapes, on arrive à isoler l'erreur : si un fichier fait 10 000 lignes, il contiendra 5 000 à la première étape, 2 500 à la deuxième, …, et 10 lignes à la dixième étape.

Si vous n'arrivez pas à résoudre le problème isolé ainsi, vous disposez d'un ECM à soumettre à la communauté Internet.

Rechercher sur Internet

Mais avant de faire appel à la communauté, il convient d'essayer de vous débrouiller par vous-même.

La première chose à faire consiste à consulter les foires aux questions (FAQ) ; vous avez de nombreuses ressources sur :

- en français :
  - http://www.gutenberg.eu.org/, le site du Groupe francophone des utilisateurs de TeX (GUT) ;
  - http://www.grappa.univ-lille3.fr/FAQ-LaTeX ;
  - http://faqfctt.fr.eu.org ;
- en anglais :
  - http://www.tex.ac.uk/faq en anglais
  - http://www.ctan.org/ (ou http://tug.ctan.org/), le *Comprehensive TeX Archive network* du TeX User Group (TUG) ;
  - http://tug.org/, le site du TUG.

Si une extension est en cause, il faut penser à consulter la documentation de cette extension ; une recherche par mots-clefs devrait vous permettre de la trouver.

Notez que vous avez également une documentation installée avec LaTeX. Sous un système Unix, vous pouvez taper en ligne de commande `texdoc ltx-2` pour avoir la documentation en HTML, et `texdoc latex2e` pour l'avoir en PDF. Si ces commandes ne fonctionnent pas, essayez de chercher « latex2e » sur votre disque dur. Avec la distribution TeXlive sous MacOS X, la documentation HTML se trouve en `/sw/share/texmf-dist/doc/help/Catalogue/index.html`, et les fichiers PDF sont dans `/sw/share/texmf-dist/doc/latex/` (il existe un répertoire pour chaque extension importante, et un répertoire `general` pour LaTeX en général) — le tout en anglais.

La deuxième chose consiste à faire une recherche avec un moteur de recherche, contenant comme mots-clefs les instruction, environnement, extension en cause, ainsi que « latex ». Une recherche dans les archives du groupe usenet fr.comp.text.tex s'impose : vous n'êtes sans doute pas le premier à avoir eu le problème, la solution s'y trouve sans doute déjà. Les archives se trouvent sur

> http://groups.google.com/group/fr.comp.text.tex

Enfin, si tout cela est infructueux, vous pouvez poster votre problème sur news:fr.comp.text.tex, (fctt pour les intimes) avec un agent usenet (rechercher « usenet » dans l'aide de votre système d'exploitation ; Microsoft Outlook remplit cette fonction, sinon, pensez à Mozilla Thunderbird).

Si vous n'avez jamais posté, la première chose consiste à configurer le lecteur de forums. vérifiez que vous postez bien en texte pur et pas en HTML (en général dans le menu **Outil | Options** ou bien le menu **Préférences** de l'application). Cherchez votre adresse mél dans la configuration, et maquillez-là pour éviter d'avoir des messages non-sollicités (spam). Puis, allez faire un test en postant sur le groupe news:fr.test.

Ceci fait, postez sur fctt :

- mettez un titre explicite (par exemple avec le nom de l'instruction, de l'environnement ou de l'extension en cause), et non pas simplement « a l'aide » ;
  ne mettez pas d'accent dans le titre ;
- dites « bonjour »,
- décrivez votre configuration (système d'exploitation, distribution de LaTeX avec son numéro de version ou sa date) ;
- décrivez votre problème, en collant le contenu de l'ECM ;
- indiquez les recherches que vous avez déjà faites.

Surveillez le forum afin de voir les réponses, on vous demandera peut-être de fournir des précisions. Si personne ne répond au bout de 2-3 jours, relancez poliment : personne n'est peut-être compétent sur votre problème, mais peut-être que quelqu'un connaît un endroit où chercher, mais attendait de voir les réponses des autres.

Si l'on vous apporte la solution, n'oubliez pas de dire merci (-:

« Protéger » les commandes « fragiles »

Dans certains cas, vous utilisez une commande de manière correcte, mais le résultat est erroné en raison du contexte dans lequel elle se trouve. Par exemple, telle commande se comporte bien dans du corps de texte, mais donnera des résultats bizarre si elle est insérée dans une commande `\chapter{…}` ou un environnement `tabular`.

Une telle commande est dite « fragile ». On peut souvent résoudre le problème en « protégeant » cette commande, c'est-à-dire en mettant une commande `\protect` devant lorsque le problème se pose.

Exemple de commandes fragiles :

- `\verb` ;
- `\numberline` ;
- `\chemical` de l'extension `ppchtex`.

Chasse aux erreurs

Avertissements fréquents

Avertissements de mise en page

`Overfull` ou `Underfull \vbox` ou `\hbox`

- Cause : le typographe virtuel a du mal à agencer le texte, ce qui peut produire un effet inesthétique ;
- prévention : aucune ;
- solution : ne pas s'en préoccuper dans un premier temps ; dans la version finale, éventuellement insérer des contraintes sur la mise en page, par exemple un saut de page avec `\newpage`.

`Float too large for page by …`

- Cause : l'objet flottant est trop grand par rapport à la page ;
- prévention : contrôler la taille des tableaux et images à intégrer ;
- solution : ajuster la taille de l'objet.

`h float specfier changed to ht`

- Cause : l'objet flottant est défini avec une position `h` *(here)*, mais il sera placé en haut de la page suivante ;
- prévention : aucune ;
- solution : faire précéder le h d'un point d'exclamation pour forcer le positionnement.

Avertissements de références

- `Label '[…]' multiply defined` ;
- `Reference '[…]' on page […] undefined` ;
- `Labels may have changed`.

Voir *Structuration du texte > Erreurs possibles*.

Erreurs fréquentes

`! No room for …`

- Cause : LaTeX gère par défaut 256 « registres » : compteurs, ressorts (blancs élastiques), blancs fixes, unités lexicales, … ; cette limite a été atteinte
- prévention : aucune ;
- solution : utiliser l'extension `etex` qui rajoute des registres supplémentaires (plus de 65 000).

Compilation en ligne de commande

Si vous compilez en ligne de commande, les messages s'affichent dans la fenêtre du *shell*, et à moins que vous ne l'ayez précisé par un option, la compilation s'interrompt à la première erreur.

Si vous ne voulez pas que la compilation s'arrête à chaque erreur, il faut la lancer avec l'option `-interaction=nonstopmode` :

```
latex -interaction=nonstopmode nom_du_fichier
```

Dans le cas contraire, la compilation peut s'arrêter et attendre une action de votre part :

- parce qu'elle ne trouve pas un fichier : il faut alors entrer le nom du fichier ;
- pour une erreur de syntaxe :
  - appuyer sur Entrée pour ignorer l'erreur et continuer,
  - taper `X` puis Entrée pour arrêter la compilation,

- `H` pour avoir des détails sur l'erreur,
- `R` pour ignorer cette erreur et toutes les suivantes,
- `?` pour connaître les possibilités.

Notes

1. il s'agit d'un texte utilisé en typographie depuis le XVI^e siècle pour faire des tests ; c'est d'un extrait du *de Finibus Bonorum et Malorum* de Cicéron, écrit en 45 av. J.-C., et dont des mots ont été tronqués, cf. *Faux-texte* et http://lipsum.com/

Voir aussi

- Doc et aide (http://www.tuteurs.ens.fr/logiciels/latex/aide.html) [archive], site des tuteurs de l'ENS
- Détecter et résoudre les problèmes (http://www.pearsoneducation.fr/Documents/Ressources/gestion-erreurs.pdf) [archive] (fichier PDF, 61 p., 464 kio), *Latex Companion 2^e édition*, F. Mittelbach, M. Goossens, D. Carlisle, C. Rowley (trad. J. André, B. Belet, J.-C. Charpentier, J.-M. Hufflen, Y. Soulet), Pearson Education

Table des matières - Généralités - Premiers pas - Structure du document - Gestion de la bibliographie - Tableaux - Images - Éléments flottants et figures - Mise en forme du texte - Choix de la police - Mise en page - Mathématiques - Gestion des gros documents - Faire des présentations - Arts et loisirs - Dessiner avec LaTeX - Créer une extension ou une classe - Programmer avec LaTeX - *Annexes* - Vade mecum - Conversion - Glossaire de typographie - Index - Commandes - Liens externes

Structure du document

Ce chapitre approfondit la notion de **structuration d'un document** abordée dans le didacticiel précédent. Le but est de vous permettre de produire un modèle d'article destiné à une publication académique.

LaTeX vous oblige pratiquement à définir la structure de votre document à l'intérieur de celui-ci. C'est tout de même une bonne chose. Parce qu'une fois que LaTeX comprend comment vous voulez organiser votre document, il prendra en charge à votre place toutes les tâches pénibles de disposition et de présentation. La séparation du contenu et de la présentation vous permet de vous concentrer sur votre travail, qui est de communiquer les résultats de vos recherches.

Avant d'expliquer les diverses commandes LaTeX, je crois qu'il serait préférable de commencer par voir à quoi ressemble le résultat final, afin que nous sachions dès le début quels seront les effets de nos actions. Jetez un coup d'œil au [tutorial2/simple.pdf] produit à partir de l'exemple de ce didacticiel. En outre, la source en LaTeX est très longue, j'emploierai donc seulement des extraits au cours de ce didacticiel. Le fichier source est disponible [tutorial2/simple.tex ici] et à la fin de cette page.

Préambule

Si vous vous rappelez du cours d'instruction précédent, le préambule est la partie se trouvant au tout début du fichier source LaTeX avant la commande `\begin{document}`. Elle contient normalement les commandes qui affectent le document entier.

```
% simple.tex – Un exemple d'article simple pour illustrer la structure d'un document.
\documentclass{article}
\usepackage[frenchb]{babel}
\usepackage[T1]{fontenc}
\usepackage{times}

\begin{document}
...
```

La première ligne est un commentaire (signalé par le symbole `%`). La commande de `\documentclass` prend un paramètre, qui est dans ce cas *article*, parce que c'est le type de document que nous voulons produire. Les autres classes existantes sont *book*, *report*, *thesis* etc. Il est également possible de créer vos propres classes, comme souvent les éditeurs de journal le font, en fournissant simplement leur propre fichier de classe qui indique à LaTeX comment composer le contenu. Mais nous nous contenterons de la classe standard d'article pour l'instant. La commande `\usepackage` est une commande importante qui demande à LaTeX d'utiliser quelques macro-commandes externes. Dans cet exemple, j'ai indiqué `times` qui signifie que LaTeX emploiera les polices postscript de type 1, qui paraissent plus jolies :).

Et enfin, le `\begin{document}` qui ne fait pas strictement partie du préambule, mais que je mettrai quand même ici, étant donné qu'il marque la fin du préambule et déclare comme son nom l'indique que le document commence à partir de cet endroit.

Voici une autre possibilité de commencer le document en plaçant l'option de langue dans le préambule :

```
% simple.tex – Un exemple d'article simple pour illustrer la structure d'un document.
\documentclass[frenchb]{article}
\usepackage{babel}
\usepackage[T1]{fontenc}

\begin{document}
...
```

ainsi chaque nouvelle commande `\usepackage` saura automatiquement (si besoin est) qu'elle doit utiliser l'option `frenchb` ici plus besoin donc de rajouter l'option `frenchb` au paquet `babel`.

À titre d'information, il existe plusieurs extensions afin d'obtenir la typographie française sous LaTeX, dont certaines sont obsolètes[1] :

- l'extension `babel` avec l'option `[frenchb]` ; l'option `[francais]` est strictement équivalente à l'option `frenchb` depuis la version 3.6 de `babel` (mais son utilisation avec une très ancienne distribution de LaTeX — d'avant 1997… — pourrait poser des problèmes) ;
- l'extension `frenchpro` de Bernard GAULLE, payante, ou `frenchle`, gratuite mais plus restreinte ;
- l'ancienne extension `french` de Bernard GAULLE est obsolète et a été remplacée par `frenchpro` et `frenchle` ; dans les anciennes versions de `babel` (avant 1997), l'appel de `french` chargeait l'extension de même nom si elle était présente, et utilisait `babel/frenchb` sinon, on avait donc un résultat qui dépendait de l'installation ; ce n'est plus le cas maintenant, `french` est devenu un équivalent de `frenchb`.

Les extensions `frenchle` et `frenchpro` semblent fournir des possibilité plus fines, notamment en ce qui concerne la gestion des guillemets (par exemple les « guillemets de suite » qui ouvrent chaque ligne d'une citation longue). Cependant, leur auteur Bernard GAULLE est mort en 2007, ce qui rend incertain la maintenance de ces extensions[2].

Bonne nouvelle ! Le paquet FrenchPro est à nouveau suivi sous le nom e-French et les modifications (il n'est plus besoin de licence) sont accessible sur le site paquets pour Windows et Linux avec sources (http://www.eFrench.org) [archive].

Entrée en matière

Au début de la plupart des documents, se trouvent des informations sur le document lui-même, telles que le titre et la date, et aussi des informations

sur les auteurs, comme leur nom, leur adresse, leur adresse courriel etc. Toutes ces informations, Latex les désigne collectivement sous le nom d'*entrée en matière*. Un exemple :

```
\title{Comment structurer un document \LaTeX{}}
\author{Andrew Roberts\\
   école d'informatique,\\
   Université de Leeds,\\
   Leeds,\\
   Royaume-Uni,\\
   LS2 1HE\\
   \texttt{andyr@comp.leeds.ac.uk}}
\date{\today}

\maketitle
```

La commande `\title` est assez simple à utiliser. Mettez simplement le titre que vous voulez entre les accolades.

La commande `\author` semble aussi assez facile d'emploi. Si vous disposez d'un fichier de classe d'un éditeur, ou si vous employez la classe de base d'article d'AMS (`amsart`), alors vous comprendrez mieux le besoin de fournir une information sur l'auteur. En attendant, vous pouvez voir comment la commande de nouvelle ligne (`\\`) est utilisée pour, comme ici, placer son adresse. Mon adresse courriel se trouve à l'extrémité, et la commande `\texttt` formate les adresses de courriel en utilisant une police de chasse fixe.

La commande de `\date` prend un paramètre pour indiquer la date de création du document. J'ai utilisé une commande intégrée appelée `\today` qui, lorsqu'elle est exécutée par Latex, est remplacée par la date du jour. Mais vous êtes libre de mettre à la place la date que vous voulez. Si les accolades sont laissées vides, alors la date est omise.

Sans `\maketitle`, l'entrée en matière n'apparaîtrait pas dans le document. Ainsi elle est nécessaire pour donner vos attributs d'article au document.

Les commandes `\title`, `\author` et `\date` définissent les attributs, la commande `\maketitle` provoque leur affichage dans une mise en forme standardisée.

Résumé

Parce que beaucoup d'articles de recherche ont un résumé, certaines classes de document disposent d'un environnement permettant de définir une partie du contenu comme étant le résumé. Celui-ci devrait apparaître dans un ordre logique, c'est-à-dire après l'entrée en matière, mais avant les sections principales du corps du document.

```
\begin{abstract}
   Votre résumé commence ici...
   ...
\end{abstract}
```

Cet environnement `abstract` est disponible pour les classes `article` et `report`, mais *pas* pour la classe `book`. Il est toutefois simple d'en bricoler un :

- ouvrir le fichier `report.cls` dans l'éditeur de texte ;
- dans ce fichier, rechercher la chaîne de caractères « abstract » ;
- copier le code trouvé dans le préambule de votre fichier, ou mieux, dans un fichier `.tex` appelé par un `\input` dans le préambule, ou encore dans une extension ou une classe personnelle si vous en avez fait une.

Commandes de sectionnement

Les commandes pour insérer des sections sont assez intuitives. Naturellement, certaines commandes sont appropriées à différentes classes de documents. Par exemple, un livre a des chapitres mais un article n'en a pas. Voici une énumération de certaines des commandes de structure utilisées dans *simple.tex*.

```
\section{Introduction}
Cette section contient...

\section{Structure}
Cette section contient...

\subsection{Entrée en matière}
Cette sous-section contient...

\subsubsection{Information de l'article}
Cette sous-sous-section contient...
```

Comme vous pouvez le voir, les commandes sont assez intuitives. Notez que vous n'avez pas besoin d'indiquer les numéros de section, Latex s'occupera de cela pour vous ! En outre, pour les sections, vous n'avez pas besoin de préciser à quel bloc appartient tel contenu, en utilisant des commandes comme `\begin` ou `\end` par exemple.

Commande	Niveau

`\part{`*`partie`*`}`	-1
`\chapter{`*`chapitre`*`}`	0
`\section{`*`section`*`}`	1
`\subsection{`*`sous-section`*`}`	2
`\subsubsection{`*`sous-sous-section`*`}`	3
`\paragraph{`*`paragraphe`*`}`	4
`\subparagraph{`*`sous-paragraphe`*`}`	5

La numérotation des sections est réalisée automatiquement par Latex, ainsi ce n'est pas la peine de rajouter explicitement de numéro ; il suffit juste d'insérer le titre que vous voulez entre les accolades. Si vous ne voulez pas de numérotation de section, ajoutez un astérisque « * » après la commande de section, mais avant la première accolade ouvrante, par exemple :

```
\section*{Un titre sans numéro}
```

dans ce cas-là, le titre ne figure pas dans le table des matières. Pour qu'il y figure, il faut ajouter la commande `\addcontentsline{toc}{`*`niveau`*`}{\protect\numberline{}`*`titre`*`}`, par exemple

```
\section*{Préface}
\addcontentsline{toc}{section}{\protect\numberline{}Préface}
```

La bibliographie

À l'instar des références croisées, les références vers des publications externes sont gérées directement par LaTeX, ce qui permet de s'affranchir des problèmes d'étiquettes (par exemple changement de numérotation si l'on introduit une nouvelle référence ou que l'on déplace du texte).

Pour plus de détails voir : ***LaTeX/Gestion de la bibliographie***.

Index

LaTeX permet de réaliser un index. Pour cela, il faut :

- utiliser l'extension `makeidx` ;
- dans le préambule, mettre la commande `\makeindex` ;
- pour créer une entrée, mettre dans le texte `\index{`*`nom de l'entrée`*`}` ;
- mettre la commande `\printindex` à l'endroit où l'on veut mettre l'index ;
- compiler en tapant `latex mon_fichier.tex` votre fichier source;
- `latex` générera en plus des fichiers habituels le fichier `mon_fichier.idx` ;
- compiler ce dernier fichier avec `makeindex` en tapant `\makeindex mon_fichier.idx` ;
- compiler une deuxième fois votre fichier source en tapant `latex mon_fichier.tex`;

L'index contiendra le *nom de l'entrée* suivi du, ou des numéros de page.

On a les possibilités suivantes :

- pour avoir un classement alphabétique ne correspondant pas au nom de l'entrée : `\index{`*`nom de classement@nom affiché`*`}`, par exemple `\index{Epee@Épée}` ;
- pour faire référence à un autre mot : `\index{`*`mot`*`|see{`*`autre mot`*`}}`, par exemple `\index{Sabre|see{Épée}}` ;
- pour avoir un intervalle de page : on place `\index{`*`mot`*`|(}` au début de la zone et `\index{`*`mot`*`|)}` à la fin ;
- pour une « cascade », on met un point d'exclamation, par exemple `\index{Lame!Sabre}`, `\index{Lame!Epee@Épée}`.

Si vous utilisez l'extension `hyperref`, les numéros de page sont des liens vers les endroits concernés.

Vous pouvez créer plusieurs index. Pour cela, utilisez l'extension `index` (l'extension `multind` est obsolète).

Pour créer un index, `makeidx` crée des fichiers intermédiaires avec le nom du document et les extensions `.idx` et `.ind`. Avec l'extension `indexw`, il faut définir de nouvelles extensions afin d'avoir plusieurs fichiers intermédiaires. Il est recommandé d'utiliser deux lettres caractéristique du nom de l'index (par exemple les deux premières), puis d'ajouter un « x » et un « d » pour avoir les deux extensions. Par ailleurs, chaque index a un nom, sa référence interne, et un titre qui sera affiché dans le document.

- appelez l'extension `index` ;
- dans le préambule, mettre les commandes `\newindex{`*`nom_de_l'index`*`}{`*`extension`*`x}{`*`extension`*`d}{`*`titre`*`}` ;
- pour créer une entrée, mettre dans le texte `\index[`*`nom_de_l'index`*`]{`*`nom de l'entrée`*`}` ;
- mettre la commande `\printindex[`*`nom_de_l'index`*`]` à l'endroit où l'on veut mettre l'index ;
- compiler chaque index en ligne de commande, avec
  `makeindex `*`document.extension`*`x -o `*`document.extension`*`d`

On peut toujours utiliser un index général, comme à l'habitude, qui se compile normalement (par exemple avec l'interface de l'éditeur de texte).

Par exemple, pour un fichier `test.tex` :

```
\documentclass{article}

\usepackage{index}
   \makeindex % index général
   \newindex{env}{enx}{end}{Environnements}
   \newindex{ext}{exx}{exd}{Extensions}
   \newindex{cmm}{cmx}{cmd}{Commandes}
\newcommand{\commande}[1]
   {\texttt{\textbackslash #1}}
\newcommand{\indexcmm}[1]
   {\index[cmm]{#1@\commande{#1}}} % index d'une commande

\begin{document}

Une citation\index{citation} hors paragraphe
se met dans un environnement
\emph{quote}\index[env]{quote}
ou \emph{quotation}\index[env]{quotation}

L'extension \emph{array}\index[ext]{array}
fournit les commandes
\commande{raggedleft}\indexcmm{raggedleft}
et \commande{raggedright}\indexcmm{raggedright}.

\printindex % index général
\printindex[env]
\printindex[ext]
\printindex[cmm]

\end{document}
```

Entre la première et la deuxième compilation du fichier `.tex`, exécuter en ligne de commande :

1. `makeindex test` (ou bien utiliser la commande `makeindex` de l'éditeur de texte) ;
2. `makeindex test.enx -o test.end` ;
3. `makeindex test.exx -o test.exd` ;
4. `makeindex test.cmx -o test.cmd`.

Notes

1. on pourra à ce titre consulter les discussions babel : french, frenchb ou francais (http://groups.google.com/group/fr.comp.text.tex/browse_thread/thread/44a3e70f2741f479/2f1c0b2fb1d4ccb8#2f1c0b2fb1d4ccb8) [archive] et Annonce changement distribution french (http://groups.google.com/group/fr.comp.text.tex/msg/b0f3975a9eb4449f) [archive] du forum fr.comp.text.tex (news:fr.comp.text.tex) [archive] (2 mars 2008)
2. voir à ce titre les discussions french ou babel ? une expérience (http://groups.google.com/group/fr.comp.text.tex/browse_thread/thread/380486bbb082e6a9) [archive] (11 novembre 2007) et Package french de Bernard GAULLE : au secours ! (http://groups.google.com/group/fr.comp.text.tex/browse_thread/thread/5a251332112ca394) [archive] (1er janvier 2008) sur le forum fr.comp.text.tex (news:fr.comp.text.tex) [archive]

Glossaire

Gestion de la bibliographie

En dehors des œuvres de fiction, tous les ouvrages devraient donner des références bibliographiques afin que le lecteur puisse vérifier les sources et prolonger sa recherche ; c'est notamment le cas des rapports de recherche ou publications académiques. La réalisation « à la main » de cette tâche est vite pénible et source d'erreur, notamment si l'on change la disposition de l'ouvrage : il faut faire attention à la numérotation, des *op. cit.* peuvent se retrouver avant la première mention de l'ouvrage, …

Heureusement, LaTeX dispose d'un outil auxiliaire appelé BibTeX (http://www.bibtex.org) [archive] qui gère automatiquement la bibliographie.

La description des ouvrages — auteurs, titre, année de parution, éditeur, … — sont écrites dans un fichier extérieur ; vous avez alors un fichier unique contenant toute votre bibliographie, et auquel se réfèrent vos différents documents LaTeX. Ainsi, si un ouvrage est mentionné dans plusieurs documents, il n'est rentré qu'une seule fois (écrire une fois, et lire beaucoup), ce qui limite les erreurs.

Vous pouvez également avoir plusieurs fichiers de bibliographie, et un document peut faire référence à plusieurs fichiers à la fois, « unifiant » ainsi les bibliographies.

L'inclusion des référence bibliographiques dans le document LaTeX nécessite l'utilisation du programme `bibtex` sur le document LaTeX.

Aperçu

Voir Structuration du texte > Bibliographie.

BibTeX

Le chapitre précédent a donné l'idée d'inclure les références à la fin du document, et d'employer la commande `\cite` afin de s'y rapporter dans le texte. Ici est présentée l'utilisation de BibTeX, plus flexible.

Une base de données de BibTeX est emmagasinée dans un fichier d'extension `.bib`. Il s'agit d'un fichier de type texte, qui peut ainsi être lu et modifié facilement. La structure d'un fichier est tout à fait simple. Voici un exemple d'une entrée de BibTeX:

```
@article{greenwade93,
    author  = "Greenwade, George D.",
    title   = "The {C}omprehensive {T}ex {A}rchive {N}etwork ({CTAN})",
    year    = "1993",
    journal = "TUGBoat",
    volume  = "14",
    number  = "3",
    pages   = "342--351"
}
```

Chaque entrée commence par la déclaration du type de référence, sous la forme `@type`. BibTeX connaît pratiquement tous les types auxquels vous pouvez penser, notamment les plus courants comme *book* pour les livres, *article* pour les articles, *inproceedings* pour les notes de conférences, etc. Dans cet exemple, je me suis référé à un article publié dans un journal.

Après le type, vous devez ouvrir une accolade `{` pour signaler le début la zone destinée aux attributs de référence. Le premier qui suit immédiatement l'accolade, correspond à une marque de citation. Cette marque doit être unique pour toutes les entrées de votre bibliographie. C'est avec ce repère que vous pourrez établir des renvois à cette entrée dans votre document. La façon de dénommer chaque référence vous appartient, mais il y a une convention très répandue qui consiste à employer le nom de famille de l'auteur, suivi de l'année de la publication. C'est cette règle que j'emploie dans ce guide.

Ensuite, il devrait vous sembler clair que les champs et les données qui suivent se rapportent à cette référence particulière. Les noms de champ à gauche sont des marques de BibTeX. Ils sont suivis par un signe d'égalité = et la valeur pour ce champ est alors placée juste après. BibTeX s'attend à ce que vous marquiez explicitement le début et la fin de chaque valeur. Il est possible d'employer les guillemets anglais `"`, ou les accolades `{`, `}`. Mais signalons que les accolades ont d'autres fonctions, dans des attributs en particulier, ainsi j'ai préféré ne pas les employer ici pour des raisons de clarté.

Rappelez-vous que chaque attribut doit être suivi d'une virgule pour qu'ils soient séparés les uns des autres. Vous n'avez pas besoin d'ajouter de virgule au dernier attribut, puisque l'accolade fermante indiquera à BibTeX qu'il n'y a plus d'attribut pour cette entrée, bien que vous n'obtiendriez aucune erreur si vous en placiez une.

Cela peut prendre un peu de temps pour apprendre ce que sont les types de référence, et quels champs sont disponibles pour chaque type (et quels sont ceux qui sont obligatoires ou facultatifs, etc.). Vous pouvez jeter un œil à la page entry type reference (http://newton.ex.ac.uk/tex/pack/bibtex/btxdoc/node6.html) [archive] ou à la page field reference (http://newton.ex.ac.uk/tex/pack/bibtex/btxdoc/node7.html) [archive] pour la description de tous les champs. Il peut être intéressant pour vous de les garder dans vos pages favorites de votre navigateur internet ou de les imprimer, de sorte qu'elles soient sous votre main quand vous en avez besoin.

Les auteurs

BibTeX gère avec souplesse les noms d'auteurs. Il peut accepter les formats de noms « `Nom, Prénom` » ou « `Prénom Nom` ». Le premier format est à préférer lorsque l'on a affaire à des prénoms multiples ou des noms composés : la virgule permet de distinguer ce qui est prénom de ce qui est nom, par exemple « `von Neumann, John` ». C'est de fait le format recommandé. Si vous préférez utilisez le format « `Prénom Nom` », alors vous devez

préciser vous-même à BibTeX de garder « von » et « Neumann » ensemble, en utilisant les accolades : « `John {von Neumann}` ».

Ensuite, il est possible de demander à *Bibtex* de faire référence à plus d'un auteur. Pour cela, il suffit simplement de placer le mot-clé **and** entre chaque auteur, comme le montre l'exemple suivant:

```
@book{goossens93,
    author    = "Goossens, Michel and Mittlebach, Frank and Samarin, Alexander",
    title     = "The Latex Companion",
    year      = "1993",
    publisher = "Addison-Wesley",
    address   = "Reading, Massachusetts"
}
```

Cette entrée spécifie que le livre a trois auteurs, et leurs noms sont séparés par des `and`. Naturellement, quand BibTeX sera exécuté il placera uniquement un « et » entre le pénultième auteur et le dernier, mais dans le fichier `.bib`, il a besoin des `and` pour différentier les auteurs.

Conservation des lettres majuscules

Par défaut, BibTeX gère lui-même les majuscules : il ne tient pas compte des majuscules et minuscules utilisées, il met des minuscules partout sauf à la première lettre du titre. Pour forcer BibTeX à préserver les majuscules, entourez la lettre concernée avec des accolades, (ou les lettres, s'il s'agit d'acronymes).

Préparer un document LaTeX afin d'utiliser votre fichier `.bib`

Cette préparation n'est pas très difficile. À la fin de votre fichier LaTeX (c'est-à-dire après le contenu, mais avant `\end{document}`), vous devez placer les commandes suivantes:

```
\bibliographystyle{plain}
\bibliography{exemple}
```

Les modèles de bibliographie sont des fichiers reconnus par BibTeX qui indiquent la façon dont les informations, emmagasinées dans le fichier `.bib`, doivent être traitées à l'exécution. Et ainsi la première commande utilisée ci-dessus déclare le nom du fichier de modèle qui doit être employé. Le fichier de modèle dans cet exemple est `plain.bst` (qui est un fichier faisant partie intégrante de BibTeX). Vous n'avez pas besoin d'ajouter l'extension `.bst` après le nom du fichier lorsque vous utilisez cette commande, car l'extension est implicite. En dépit de son nom, le modèle *plain* remplit parfaitement sa fonction (regardez le résultat figurant dans cette page pour vous rendre compte de ce que je veux dire).

La deuxième commande est celle qui indique réellement le fichier `.bib` que vous souhaitez employer. Celui que j'ai créé pour ce cours a été nommé `exemple.bib`, mais de nouveau, vous n'avez pas besoin d'ajouter l'extension du fichier. Il est supposé pour l'instant que le fichier `.bib` se trouve dans le même répertoire que le document LaTeX. Cependant, si votre fichier `.bib` se trouvait ailleurs (ce qui peut être envisagé si vous avez l'intention de mettre en place une base de données centralisée contenant toutes les références utiles pour tout votre travail), vous devriez préciser le chemin d'accès à votre fichier, par exemple

```
\bibliography{$HOME/quelque/part/exemple.bib}
```

Maintenant que le LaTeX et BibTeX savent où trouver les fichiers appropriés, il est assez simple de citer des références. La commande `\cite{`*`clef_de_référence`*`}` peut être appelée en vous assurant que l'entrée *clef_de_référence* figure bien dans le fichier `.bib`. Si vous souhaitez citer plus d'une référence à la fois, faites comme suit: `\cite{`*`clef_de_référence 1, clef_de_référence 2, ..., clef_de_référence 3`*`}`.

Pourquoi LaTeX ne génère-t-il aucun fichier ?

L'utilisation de BibTeX introduit des complications supplémentaires dans le traitement du fichier source. En effet, le fichier source écrit en LaTeX, renferme des références à des citations contenues dans la base de données figurant dans un autre fichier.

Ces complications sont en grande partie dissimulées à l'utilisateur, mais en raison de toute la complexité de la gestion des références, il est nécessaire à LaTeX d'effectuer plusieurs passages pour accomplir cette charge. Cela signifie que vous devez exécuter LaTeX un certain nombre de fois, et à chaque passage, il effectuera une tâche particulière jusqu'à ce qu'il soit parvenu à résoudre toutes les références de citation. Voici ce que vous devez taper au clavier :

1. `latex fichier_principal` (extension `.tex` inutile)
2. `bibtex fichier_principal` (extension `.aux` inutile)
3. `latex fichier_principal`
4. `latex fichier_principal`

Après avoir exécuté LaTeX une première fois, vous pourrez voir s'afficher des avertissements tels que :

```
LaTeX Warning: Citation `lamport94' on page 1 undefined on input line 21.
...
LaTeX Warning: There were undefined references.
```

L'étape suivante est l'exécution de BibTeX sur ce même fichier source en LaTeX (et non pas sur le fichier `.bib` actuel) pour définir alors toutes les références figurant dans le document. Vous devriez voir s'afficher des messages de la forme :

```
This is BibTeX, Version 0.99c (Web2C 7.3.1)
The top-level auxiliary file: fichier_principal.aux
The style file: plain.bst
Database file #1: sample.bib
```

La troisième étape, est l'exécution de LaTeX pour la deuxième fois, ce qui aura pour effet d'afficher encore plus de messages d'avertissements comme « `LaTeX Warning: Label(s) may have changed. Rerun to get cross-references right.` ». Ne vous inquiétez pas, l'opération est pratiquement terminée. Comme vous pouvez le deviner, tout ce qu'il vous reste à faire est de suivre ses instructions, et d'exécuter LaTeX pour la troisième fois, et le document sera produit comme prévu, sans aucun autre problème.

Personnalisation de l'aspect de la bibliographie

Un des principaux avantages de BibTeX, et particulièrement pour des personnes qui écrivent beaucoup de rapports de recherche, est la possibilité d'adapter la bibliographie afin de satisfaire aux exigences d'un éditeur donné. Vous noterez que différents éditeurs tendent à avoir leur propre modèle de mise en page des références, auquel les auteurs doivent se conformer s'ils veulent que leur manuscrit soit publié. En fait, les revues importantes et souvent les organisateurs de conférence ont créé leur propre modèle de bibliographie (fichier d'extension .bst) destiné aux utilisateurs de BibTeX, afin de libérer l'auteur de ce dur travail.

Il est possible de réaliser cela en raison de la nature de la base de données du fichier d'extension .bib, dans lequel toutes les informations sur vos références sont emmagasinées dans un format structuré, mais rien concernant le style. C'est un thème récurrent dans le système LaTeX en général, qui essaye autant que possible de maintenir le contenu et la présentation séparés, comme il se doit !

Un fichier de modèle de bibliographie (`.bst`) indiquera à LaTeX comment composer chaque attribut, dans quel ordre les placer, quelle ponctuation employer entre chaque attributs particuliers etc.

Malheureusement, créer un tel modèle à la main n'est pas une tâche évidente. C'est la raison pour laquelle l'outil `Makebst` (aussi appelé *custom-bib*) va vous être d'un grand secours.

`Makebst` peut être employé pour générer automatiquement un fichier .bst en respectant vos besoins. Il est très simple d'utilisation et vous posera une série de questions sur vos préférences. Ensuite, il produira le fichier modèle approprié que vous pourrez employer.

Il devrait être déjà installé avec la distribution de LaTeX (sinon, il ne vous reste plus qu'à le télécharger (http://www.mps.mpg.de/software/latex/localtex/localltx.html#makebst) [archive]) et il est très facile ensuite de l'exécuter. À l'invite de commande, tapez

```
latex makebst
```

LaTeX trouvera le fichier approprié et l'interrogatoire débutera. Vous devrez répondre à un certain nombre de questions (en remarquant que les réponses par défaut sont assez cohérentes), mais il serait compliqué de donner un exemple dans ce guide. Son utilisation est assez simple, mais si vous avez besoin d'aide, vous pouvez consulter le manuel (http://www.mps.mpg.de/software/latex/localtex/doc/merlin.pdf) [archive] complet disponible en ligne (en anglais). Je recommanderais d'expérimenter le logiciel sur un document LaTeX et de voir en fonction des résultats obtenus.

Si vous employez un fichier personnalisé .bst, il est important que LaTeX puisse le trouver ! Ainsi, assurez-vous qu'il est dans le même répertoire que le fichier source en LaTeX, à moins que vous utilisiez l'un des fichiers de modèle standard tels que *plain* ou *plainnat* qui accompagnent LaTeX, auquel cas ils seront automatiquement trouvés dans les répertoires où ils sont installés.

En outre, assurez-vous que le nom du fichier `.bst` que vous voulez employer figure bien dans la commande `\bibliographystyle{style}` (mais pas avec l'extension `.bst`!).

Quelques exemples supplémentaires

Ci-dessous, vous trouverez quelques exemples supplémentaires d'entrées de bibliographie. Le premier couvre le cas d'auteurs multiples en employant le format **nom de famille**, **Prénom** et le deuxième illustre le cas de « incollection ».

```
@article{AbedonHymanThomas2003,
  author = "Abedon, S. T. and Hyman, P. and Thomas, C.",
  year = "2003",
  title = "Experimental examination of bacteriophage latent-period evolution as a response to bacterial availability",
  journal = "Applied and Environmental Microbiology",
  volume = "69",
  pages = "7499-7506"
}

@incollection{Abedon1994,
  author = "Abedon, S. T.",
  title = "Lysis and the interaction between free phages and infected cells",
  pages = "397-405",
  booktitle = "Molecular biology of bacteriophage T4",
  editor = "Karam, Jim D. Karam and Drake, John W. and Kreuzer, Kenneth N. and Mosig, Gisela
            and Hall, Dwight and Eiserling, Frederick A. and Black, Lindsay W. and Kutter, Elizabeth
```

```
            and Carlson, Karin and Miller, Eric S. and Spicer, Eleanor",
  publisher = "ASM Press, Washington DC",
  year = "1994"
}
```

Obtenir des données bibliographiques

Plusieurs bases de données en ligne fournissent des données bibliographiques au format BibTeX, facilitant la construction de votre propre base de données. Par exemple, Google Scholar (http://scholar.google.com) [archive] dispose d'une option permettant de renvoyer les données dans un format approprié, mais vous devez changer les préférences dans Preferences (http://scholar.google.de/scholar_preferences?hl=en&lr=&output=search) [archive]. Voici un exemple d'une telle entrée BibTeX: http://scholar.google.de/scholar.bib?hl=en&lr=&q=info:qg-r7UNl9tYJ:scholar.google.com/&output=citation&oe=ASCII&oi=citation

Outils utiles

Zotero (http://www.zotero.org/) [archive] est un plugiciel Firefox (il existe également sous forme d'application) permettant la création ou l'import de données bibliographiques (à partir de la BNF par exemple). Il permet également l'export dans différents format dont BibTex.

JabRef (http://jabref.sourceforge.net/) [archive] est un petit programme en Java qui vous permet de modifier facilement vos fichiers BibTeX et d'autres bases de données bibliographiques, vous laissant (la plupart du temps) oublier les détails.

BibDesk (http://bibdesk.sourceforge.net/) [archive] est un gestionnaire de bibliographie très complet pour MacOSX.

bibliographer (http://bibliographer.homelinux.net/) [archive] Le bibliographe est un rédacteur de base de données de bibliographie au format BibTeX qui vise à être facile d'emploi. Il permet de lier des fichiers à vos enregistrements au moyen d'un moteur d'indexation et de recherche. L'interface est conçue pour un parcours aisé de votre bibliographie, et le double cliquètement sur un enregistrement ouvrira le fichier lié.

cb2Bib (http://www.molspaces.com/cb2bib/) [archive] Le cb2Bib est un outil pour extraire rapidement des références bibliographiques sans format particulier et non normalisées de messages électroniques d'alerte, de pages d'un journal internet et de fichiers au format pdf.

KBibTeX (http://www.unix-ag.uni-kl.de/~fischer/kbibtex/) [archive] KBibTeX est un rédacteur, sous KDE, de fichiers de bibliographie au format BibTeX, utilisés avec LaTeX. Il dispose de masques d'entrée confortables, permet des requêtes de recherche sur internet (par exemple avec Google Scholar ou PubMed) et exporte vers différents formats PDF, PostScript, RTF et XML/HTML. Comme KBibTeX utilise la technologie KParts de KDE, il peut s'intégrer dans Kile ou Konqueror.

Bibwiki (http://www.plaschg.net/bibwiki) [archive] Bibwiki est une SpecialPage pour MediaWiki dans laquelle vous pouvez gérer vos fichiers BibTeX. Il offre un moyen simple d'importer et exporter des enregistrements bibliographiques.

Bibliographie sans BibTeX

Même sans BibTeX, LaTeX a une approche légèrement plus intelligente du contrôle de vos références que propose une unité de traitement de texte habituelle, où tout doit être entré manuellement (à moins que vous n'achetiez une extension).

Il y a deux étapes pour installer vos bibliographie et références dans un document. La première chose est d'installer un environnement de bibliographie, qui doit se trouver à l'endroit où vous fournissez à LaTeX les détails des références. La seconde chose est de citer vos références dans votre document. Le code suivant a été utilisé à l'origine pour créer l'environnement de bibliographie du document, dans ce cours d'instruction. Il est situé juste après la dernière ligne du contenu de document, mais avant la commande de `\end{document}`.

```
\begin{thebibliography}{9}
```

```
  \bibitem{lamport94}
        Leslie Lamport,
        \emph{\LaTeX: A Document Preparation System}.
        Addison Wesley, Massachusetts,
        2nd Edition,
        1994.
```

```
\end{thebibliography}
```

Bon, qu'allons-nous voir maintenant ? La première chose à remarquer est que dans l'ouverture d'un environnement de bibliographie avec le mot-clé `thebibliography`, LaTeX considère tout ce qui se trouve entre le début et la fin des balises comme des données pour la bibliographie.

Le deuxième paramètre de l'environnement `thebibliography` (ici {9}) détermine la largeur maximale des étiquettes, et ainsi la largeur de la colonne des étiquettes. Ce qui compte ici n'est pas le chiffre 9 (on peut mettre 7 à la place par exemple), mais que c'est un seul chiffre. Ainsi on annonce à LaTeX qu'il ne va pas y avoir des étiquettes à deux ou plusieurs chiffres. Ainsi, si on a entre 10 et 99 entrés avec numérotation automatique, à la place de 9 il va falloir mettre un nombre à deux chiffres (par exemple 99). Si on n'utilise pas une numérotation automatique des entrés, mais des étiquettes personnalisés, comme [Lamp94] par exemple, dans ce cas à cette endroit il va falloir mettre l'étiquette la plus large.

Vient ensuite l'entrée de la référence proprement dite. Elle est précédée aussi de la commande `\bibitem{`*`clé_de_citation`*`}`. *clé_de_citation* devrait être un identificateur unique pour cette référence particulière, et est souvent une sorte de mnémonique se composant d'une suite de lettres,

de chiffres et de symboles de ponctuation (mais pas de virgule). J'emploie souvent le nom de famille du premier auteur, suivi des deux derniers chiffres de l'année (par conséquent *lamport94*). Si cet auteur a produit plus d'une référence pendant une année donnée, alors j'ajoute des lettres juste après, « a », « b », etc. Mais, vous devriez savoir mieux que moi ce qui vous convient le mieux. Tout ce qui suit la clef est la référence elle-même. Vous devez la dactylographier exactement comme vous voulez qu'elle soit présentée. J'y ai mis les différentes parties de la référence, telles que l'auteur, le titre, etc., sur différentes lignes pour la lisibilité. Ces marques de fin de ligne sont ignorées par LaTeX. J'ai voulu que le titre soit en italiques, ainsi j'ai employé la commande `\emph{}` afin de réaliser ceci.

Si on ne souhaite pas une numérotation automatique des entrés, on peut ajouter un paramètre optionnel à `\bibitem`, par exemple `\bibitem[Lamp94]{lamport94}`, qui sera utilisé à la place du nombre.

Il est très facile de citer un ouvrage donné. Il suffit de vous positionner à l'endroit où vous voulez que la citation apparaisse, et d'y placer : `\cite{`*`clé_de_citation`*`}`, où *clé_de_citation* est la clef de l'entrée de la référence de l'ouvrage que vous souhaitez citer. Quand LaTeX traite le document, la citation sera liée aux entrées dans la bibliographie par des références croisées et remplacée par la citation correspondant à ce nombre. De nouveau, l'avantage ici, est que LaTeX s'occupe de la numérotation pour vous. Si tout était à faire manuellement, alors ajouter ou enlever une référence serait une vraie corvée, puisque vous devriez numéroter à nouveau toutes les citations à la main.

Naturellement, vous préféreriez peut-être utiliser un système de mise en référence différent, tel que Harvard, au lieu du système numérique par défaut. Ceci sera traité dans ces pages plus tard, à moyen terme. Pourquoi n'essayeriez-vous pas de découvrir paquet Natbib (http://www.act.cmis.csiro.au/gjw/tex/docs/natbib.pdf) [archive].

Résumé

Bien qu'il y ait besoin d'un peu de temps pour maîtriser BibTeX, à long terme il est un outil efficace permettant de manipuler vos références bibliographiques. Il n'est pas rare de trouver des fichiers `.bib` sur les sites internet que les gens utilisent pour leur propres publications, ou des aperçus de travaux personnels dans un domaine particulier, etc. Dans ces énormes bases de données de bibliographie en ligne, vous trouverez souvent des versions BibTeX de publications, ainsi il est facile d'effectuer un rapide copier-coller de ces données dans votre propre fichier `.bib`, et pas de soucis !

Avoir toutes vos références dans un seul endroit peut être un grand avantage, et les avoir alors toutes sous une forme structurée permettant la personnalisation du résultat final, en est un autre. Il existe une variété d'utilitaires libres qui peuvent ouvrir vos fichiers `.bib`, et vous permettent de les visualiser d'une façon plus efficace, et aussi de les trier, ou chercher les erreurs.

Voir aussi

Liens externes


Literatur-Generator

- BibLaTeX (http://www.ctan.org/tex-archive/macros/latex/exptl/biblatex/) [archive] (site du CTAN) :
  L'extension `biblatex` est une réécriture de la gestion de la bibliographie proposée par LaTeX en conjonction avec BibTeX. Elle redéfinit la manière dont LaTeX interagit avec BibTeX à un niveau relativement fondamental. Avec `biblatex`, BibTeX ne sert qu'à classer la bibliographie et générer les étiquettes. La mise en forme n'est pas contrôlée par des fichiers de style BibTeX, mais entièrement par des macros TeX. Pour créer des nouveaux styles de bibliographie et de référence, il suffit de posséder une bonne connaissance opérationnelle de LaTeX. Il est inutile d'apprendre le langage postfixé de BibTeX. […]
  L'extension est entièrement régionalisable et peut être interfacée avec l'extension `babel`.
  (Extrait de la traduction du fichier `readme`.)
- (anglais) BibTeX Style Examples (http://www.cs.stir.ac.uk/~kjt/software/latex/showbst.html) [archive] : manière dont est composée la table des références selon le style choisi
- allemand http://www.literatur-generator.de/

Tableaux

Dans les livres scolaires, les tableaux sont souvent utilisés pour récapituler des résultats d'une recherche. Il est donc nécessaire de maîtriser leur emploi afin de produire des documents de bonne qualité.

La gestion des tableaux n'est cependant pas très intuitive. Les tableaux de base ne sont pas très astreignants, leur logique est similaire au HTML (cf. *Programmation HTML/Tableaux*), mais vous remarquerez assez rapidement qu'un tableau un peu plus élaboré demande plus de temps pour sa construction.

Avant de continuer, veillez à bien avoir assimilé les tableaux de base présentés dans la section *Premiers pas : faire des tableaux*.

L'environnement *tabular*

Rappelons quelques concepts déjà explicités.

Environnement

Un environnement est une déclaration particulière destinée à la composition du texte dans un style spécifique. Tous les environnements commencent et finissent de la même façon :

```
\begin{nom-environnement}
...
...
\end{nom-environnement}
```

Environnement `tabular`

L'environnement `tabular` est un autre type d'environnement, conçu pour disposer vos données dans des tableaux. Des paramètres sont requis après la déclaration de l'environnement pour décrire l'alignement de chaque colonne. Le nombre de colonnes n'a pas besoin d'être indiqué puisqu'il est déduit du nombre de paramètres fournis. Il est également possible d'ajouter des traits verticaux entre les colonnes. Les symboles suivants sont disponibles pour décrire les colonnes du tableau :

- `l` : colonne alignée à gauche
- `c` : colonne centrée
- `r` : colonne alignée à droite
- `p{`*`largeur`*`}` : colonne de largeur fixée, justifiée et avec alinéa ; le texte est positionné en haut de la cellule
- `m{`*`largeur`*`}` : comme précédemment, mais le texte est centré verticalement
- `b{`*`largeur`*`}` comme précédemment, mais le texte est positionné en bas de la cellule

Notez bien que l'alignement vertical se fait par rapport *aux lignes de texte* : première ligne pour `p`, dernière ligne pour `b`, centre du paragraphe pour `m`. Cela peut poser problème lorsqu'une case contient une image : l'alignement ne se fait pas par rapport à la hauteur de l'image mais pas rapport à la ligne sur laquelle elle est posée, donc par rapport à la ligne unique posée au fond de la cellule. On peut contourner ce problème avec un `\raisebox`.

Remarque : les paramètres `m` et `b` nécessitent l'utilisation de l'extension `array`

- `|` : filet vertical
- `||` : double filet vertical

Une fois dans l'environnement,

- `&` : séparateur de colonne
- `\\` : débute une nouvelle ligne
- `\hline` : filet horizontal

Remarquez, que les espaces insérés entre ces commandes sont purement inutiles, mais ils facilitent la lecture.

Tableau de base

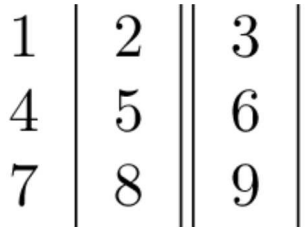
Cet exemple montre comment créer un simple tableau en LaTeX. C'est un tableau trois par trois, mais sans aucun filet.

```
\begin{tabular}{ l c r }
   1 & 2 & 3 \\
   4 & 5 & 6 \\
   7 & 8 & 9 \\
 \end{tabular}
```

1	2	3
4	5	6
7	8	9

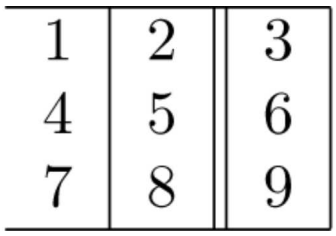
Modifions celui-ci en y ajoutant quelques filets verticaux, simples ou doubles :

```
\begin{tabular}{ l | c || r | }
   1 & 2 & 3 \\
   4 & 5 & 6 \\
   7 & 8 & 9 \\
 \end{tabular}
```



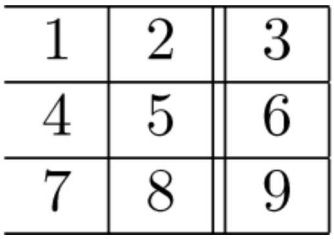
Pour ajouter des filets horizontaux tout en haut et tout en bas du tableau :

```
\begin{tabular}{ l | c || r | }
 \hline
   1 & 2 & 3 \\
   4 & 5 & 6 \\
   7 & 8 & 9 \\
 \hline
 \end{tabular}
```



Et finalement, pour ajouter des filets centrés entre toutes les lignes. Remarquez l'utilisation de l'environnement `center` pour centrer le tableau dans la page. On peut aussi obtenir un filet double horizontal en doublant la commande `\hline` (non représenté ici).

```
\begin{center}
   \begin{tabular}{ l | c || r | }
     \hline
     1 & 2 & 3 \\ \hline
     4 & 5 & 6 \\ \hline
     7 & 8 & 9 \\
     \hline
   \end{tabular}
 \end{center}
```



Texte dans un tableau

Les algorithmes de LaTeX pour mettre en forme les tableaux ont quelques imperfections. L'une d'entre elles est qu'il ne mettra pas automatiquement le texte sur plusieurs lignes dans les cellules, même si celui-ci déborde de la largeur de la page. Pour les colonnes qui contiendront une certaine quantité de texte, il est recommandé d'employer l'attribut `p` et d'indiquer la largeur désirée de la colonne (bien que cela puisse obliger à effectuer quelques ajustements avant d'obtenir le résultat escompté).

Avant de procéder, nous devons présenter le système de mesure que LaTeX emploie. Il est très souple puisque vous pouvez choisir parmi toute une variété d'unités de longueur :

- `pt` : point anglo-saxon, 1/72 de pouce ;
- `mm` : millimètre ;
- `cm` : centimètre ;
- `in` : pouce (2,54 cm) ;
- `ex` : hauteur d'x (parfois appelé à tort hauteur d'œil), hauteur d'une lettre sans hampe ni jambage dans la police courante ;
- `em` : cadratin, grossièrement la largeur d'un **M** (capitale) dans la police courante.

Il existe des commandes connues sous le nom de *commandes de longueur*, qui jouent le rôle de variable, qui n'ont pas de valeurs fixes car elles dépendent de la configuration de la classe et/ou du préambule courants du document. Les plus utiles sont :

- `\parindent` : la taille de l'indentation ;
- `\baselineskip` : distance verticale entre les lignes ;
- `\parskip` : espace supplémentaire entre les paragraphes ;
- `\textwidth` : la largeur d'une ligne de texte dans l'environnement local (par exemple, les lignes sont généralement plus étroites dans le résumé que dans le texte normal) ;
- `\textheight` : la hauteur du texte dans la page ;

Les exemples fournis sont très longs parce que j'illustrais ce qui se produit quand il y a juste un morceau de texte dans les cellules d'un tableau. Ainsi au lieu de le reproduire dans la page, allez ([15] (http://www.andy-roberts.net/misc/latex/latextutorial4.html)) directement consulter le fichier Latex en exemple, wrapped.tex (http://www.andy-roberts.net/misc/latex/tutorial4/wrapped.tex) [archive] et puis regardez le résultat (http://www.andy-roberts.net/misc/latex/tutorial4/wrapped.pdf) [archive].

Alignement du texte

Lorsque l'on utilise des colonnes du type p, m, c ou b, il peut être utile de changer la façon dont le paragraphe est aligné. Pour cela la méthode la plus claire est de définir un nouveau type de colonne qui réalise l'alignement désiré. Par exemple le code latex suivant définit un nouveau type de colonne ("M") qui centre verticalement le paragraphe par rapport à la ligne (comme le type de colonne m) et aligne le paragraphe à gauche de la colonne ("drapeau droit" en typographie, `\raggedright` en LaTeX) :

```
\newcolumntype{M}[1]{>{\raggedright}m{#1}}
```

On peut ensuite l'utiliser dans un tableau comme suit :

```
\begin{tabular}{|l|M{4cm}|}
    \hline
    colonne 1 & colonne "M" \tabularnewline
    \hline
    court texte & Texte plus long qui sera centré dans la ligne et aligné à gauche dans la colonne  \tabularnewline
    \hline
 \end{tabular}
```

Ce qui donne :

colonne 1	colone "M"
court texte	Texte plus long qui sera centré dans la ligne et aligné à droite dans la colonne

À noter :

- Cet exemple nécessite l'utilisation de l'extension `array` (`\usepackage{array}`)
- L'utilisation de `\tabularnewline` à la place de `\\` pour terminer une ligne du tableau.

Pour avoir du texte centré verticalement et horizontalement, il faut utiliser

```
\newcolumntype{M}[1]{>{\centering\arraybackslash}m{#1}}
```

D'autres solutions existent, voir cette page (http://www.latex-community.org/viewtopic.php?f=5&t=1560) [archive] (en anglais).

Définition groupée de colonnes

Il est possible de définir les colonnes identiques ou les groupes de colonnes identiques d'un tableau en une seule fois. Cela est particulièrement utile pour les tableaux contenant beaucoup de colonnes identiques ou des tableaux pour lesquels la syntaxe des colonnes est complexe.

La syntaxe est la suivante : `*{nombre}{déclaration}`

Voici un exemple de tableau à 10 colonnes :

```
\begin{tabular}{|*{10}{c|}}
    \hline
     1  & 2  & 3  & 4  & 5  & 6  & 7  & 8  & 9  & 10 \\
    \hline
     2  & 4  & 6  & 8  & 10 & 12 & 14 & 16 & 18 & 20 \\
    \hline
     3  & 6  & 9  & 12 & 15 & 18 & 21 & 24 & 27 & 30 \\
    \hline
     4  & 8  & 12 & 16 & 20 & 24 & 28 & 32 & 36 & 40 \\
    \hline
     5  & 10 & 15 & 20 & 25 & 30 & 35 & 40 & 45 & 50 \\
    \hline
     6  & 12 & 18 & 24 & 30 & 36 & 42 & 48 & 54 & 60 \\
    \hline
     7  & 14 & 21 & 28 & 35 & 42 & 49 & 56 & 63 & 70 \\
    \hline
     8  & 16 & 24 & 32 & 40 & 48 & 56 & 64 & 72 & 80 \\
    \hline
     9  & 18 & 27 & 36 & 45 & 54 & 63 & 72 & 81 & 90 \\
    \hline
\end{tabular}
```

Résultat :

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90

L'environnement `tabular*`, maîtrise de la largeur d'un tableau

C'est fondamentalement une légère extension de la version originale de tableau, bien qu'elle exige un paramètre supplémentaire (avant les descriptions de colonne) pour indiquer la largeur souhaitée du tableau.

```
\begin{tabular*}{0.75\textwidth}{ | c | c | c | r | }
  \hline
  label 1 & label 2 & label 3 & label 4 \\
  \hline
  item 1  & item 2  & item 3  & item 4  \\
  \hline
\end{tabular*}
```

Cependant, cela ne ressemble pas à ce qui était attendu. Les colonnes ont toujours leur largeur normale (juste assez large pour adapter leur contenu) tandis que les lignes sont aussi larges que la largeur demandée du tableau.
Le tableau a une apparence très laide. La raison de ce désordre est que vous devez également insérer un espace supplémentaire de colonne. Heureusement, Latex a des longueurs élastiques, qui à la différence d'autres, ne sont pas fixes, et Latex peut dynamiquement décider du moment jusqu'auquel elles doivent rester fixes. Ainsi, la solution est :

```
\begin{tabular*}{0.75\textwidth}{@{\extracolsep{\fill}} | c | c | c | r | }
  \hline
  label 1 & label 2 &  3 & label 4 \\
  \hline
  item 1  & item 2  &  3  & item 4  \\
  \hline
\end{tabular*}
```

Vous remarquerez la présence de la construction @{...} ajoutée au début de la description de colonne. Plus de détails de ce rajout seront donnés sous peu. À l'intérieur de celle-ci, la commande \extracolsep, exige une largeur comme paramètre. Une largeur fixe aurait pu être employée, cependant, en utilisant une longueur élastique, telle que \fill, les colonnes sont automatiquement espacées de tel sorte que le résultat voulu est atteint.

L'environnement `tabularx` - Calcul automatique de la largeur des colonnes

Le paquet `tabularx` met à disposition l'environnement `tabularx` permettant de définir des colonnes de type `X` qui adaptent automatiquement leur largeur de sorte que le tableau ait la largeur souhaitée. L'environnement `tabularx` prend comme argument la largeur du tableau puis la déclaration des colonnes du tableau :

```
\begin{tabularx}{largeur}{déclaration}
\end{tabularx}
```

Les déclarations des colonnes sont identiques à celles d'un tableau classique sauf le type `X` dont la largeur s'adapte pour que le tableau ait la largeur indiquée. Voici un exemple tiré en partie de "The LateX companion" p113.

```
\begin{tabularx}{15cm}{|c|p{4cm}|X|}
    \hline
    1 & The two gentlemen of Verona & The Taming of the Shrew \\
    \hline
    2 & The comedy of errors & Love's Labour's Lost \\
    \hline
    3 & The Merchant of venice & The Merry Wives of Windsor \\
    \hline
\end{tabularx}
```

Voici le résultat, on remarque le comportement différent de la colonne `p` et de la colonne `X` :

1	The two gentlemen of Verona	The Taming of the Shrew
2	The comedy of errors	Love's Labour's Lost
3	The Merchant of venice	The Merry Wives of Windsor

Espacement entre les lignes, espacement entre les colonnes

Pour aérer un tableau, il est possible de redéfinir l'espacement entre les lignes d'un tableaux et l'espacement entre les colonnes, par exemple :

\renewcommand{\arraystretch}{1.5}

\setlength{\tabcolsep}{1cm}

Expressions avec @

Maintenant qu'il a été présenté, il serait intéressant de nous pencher sur le « prescripteur » @, bien que celui-ci n'ait pas un grand nombre d'applications évidentes.

Il prend typiquement du texte comme paramètre, et lorsqu'il est apposé à une colonne, il insère automatiquement ce texte dans chaque cellule de cette colonne avant les données mêmes de cette cellule. Il faut préciser qu'une fois utilisé, l'espace entre les colonnes concernées est supprimé. Pour ajouter un espace, employez la commande @{\hspace{*largeur*}}.

Pour mieux comprendre cela exigera quelques exemples d'utilisation. Parfois, il est souhaitable dans les tables scientifiques d'avoir les nombres alignés sur la virgule décimale. Ceci peut être réalisé en faisant ce qui suit :

```
\begin{tabular}{r@{,}l}
  3&14159\\
  16&2\\
  123&456\\
\end{tabular}
```

Son application supprimant les espaces, il peut être très utile pour maîtriser l'espacement horizontal entre les colonnes. À partir d'un tableau de base, changeons les descriptions des colonnes:

```
\begin{tabular}{|l|l|}
\hline
truc & truc \\ \hline
truc & truc \\
\hline
\end{tabular}
```

{|l|l|}

truc	truc
truc	truc

{|@{}l|l@{}|}

truc	truc
truc	truc

{|@{}l@{}|l@{}|}

truc	truc
truc	truc

{|@{}l@{}|@{}l@{}|}

truc	truc
truc	truc

Lignes enveloppant des colonnes multiples

La commande adéquate ressemble à ceci:

```
\multicolumn{nomb_col}{alignement}{contenu}.
```

- *nomb_col* est le nombre de colonnes consécutives à fusionner;
- *alignement* est assez évident, et doit être remplacé par `l`, `c`, ou `r` ;
- *contenu* représente simplement les données mêmes que vous désirez inclure dans la cellule.

Donnons un exemple simple:

```
\begin{tabular}{|l|l|}
  \hline
  \multicolumn{2}{|c|}{Membres de l'équipe} \\
  \hline
  GK & Guillaume Warmuz \\
  LB & Yohan Lachor \\
  DC & Jean-Guy Wallemme \\
  DC & Cyrille Magnier \\
  RB & Eric Sikora \\
  MC & Marc-Vivien Foe \\
  MC & Stephane Ziani \\
  MC & Mickael Debeve \\
  FW & Tony Vairelles \\
  ST & Vladimir Smicer \\
  ST & Matt Moussilou \\
  \hline
\end{tabular}
```

Membres de l'équipe	
GK	Guillaume Warmuz
LB	Yohan Lachor
DC	Jean-Guy Wallemme
DC	Cyrille Magnier
RB	Eric Sikora
MC	Marc-Vivien Foe
MC	Stephane Ziani
MC	Mickael Debeve
FW	Tony Vairelles
ST	Vladimir Smicer
ST	Matt Moussilou

Colonnes enveloppant des lignes multiples

Nous avons proposé précédemment une méthode simple mais détournée (cf. *Faire des tableaux > Fusionner les lignes*).

Si l'on veut vraiment fusionner les lignes, il faut avoir recours à l'extension `multirow`, et donc ajouter

```
\usepackage{multirow}
```

dans le préambule. Cela fournit alors la commande requise pour envelopper des lignes :

```
\multirow{nomb_ligne}{largeur}{contenu}
```

Les paramètres sont assez simples à comprendre. Avec le paramètre *largeur*, vous pouvez indiquer une largeur fixe si vous le désirez, ou une largeur normale (c'est-à-dire, juste assez large pour adapter le contenu de la colonne) en entrant simplement un astérisque (*). Cette approche a été employée dans l'exemple suivant:

```
\begin{tabular}{|l|l|l|}
  \hline
  \multicolumn{3}{|c|}{Membres de l'équipe} \\
  \hline
  Gardien de but & GK & Paul Robinson \\ \hline
  \multirow{4}{*}{Défenseurs} & LB & Lucus Radebe \\
    & DC & Michael Duberry \\
    & DC & Dominic Matteo \\
    & RB & Didier Domi \\ \hline
  \multirow{3}{*}{Milieux de terrain } & MC & David Batty \\
    & MC & Eirik Bakke \\
    & MC & Jody Morris \\ \hline
  Avant & FW & Jamie McMaster \\ \hline
  \multirow{2}{*}{Attaquants} & ST & Alan Smith \\
    & ST & Mark Viduka \\
 \hline
\end{tabular}
```

Membres de l'équipe		
Gardien de but	GK	Paul Robinson
Défenseurs	LB	Lucus Radebe
	DC	Michael Duberry
	DC	Dominic Matteo
	RB	Didier Domi
Milieux de terrain	MC	David Batty
	MC	Eirik Bakke
	MC	Jody Morris
Avant	FW	Jamie McMaster
Attaquants	ST	Alan Smith
	ST	Mark Viduka

Ce qui est essentiel de remarquer lorsque vous utilisez la commande \multirow sur les lignes consécutives à envelopper, est qu'une entrée vide devant les cellules concernées doit être insérée : en effet, la commande `\multirow` indique simplement que le *texte* est sur plusieurs lignes, il ne modifie pas la structure du tableau (contrairement à `\multicolumn`). Ceci a de l'importance lorsque l'on met un fond coloré (voir ci-après).

Couleur

Nous avons vu dans *Programmation LaTeX/Options de mise en forme avancées > Couleur* comment mettre du texte en couleur, caractères et fond.

Mais le fond du texte est une boîte plus petite que la cellule du tableau. Si l'on veut mettre le fond d'une cellule en couleur, il faut utiliser l'extension `colortbl`. Vous pouvez l'appeler en même temps de `xcolor` en utilisant l'option `table` :

```
\usepackage[table]{xcolor}
```

(si vous utilisez beamer, vous devriez aussi avoir : `\documentclass[xcolor=table]{beamer}`)

Le recours à l'extension `array` est fortement recommandé pour améliorer le rendu.

Vous pouvez alors définir :

- la couleur de fond d'une ligne en mettant `\rowcolor{`*`couleur`*`}` en début de ligne ;
- la couleur de fond d'une colonne en mettant `>{\columncolor{`*`couleur`*`}}` avant la désignation de la colonne dans la définition de l'environnement `tabular` ; on peut lui adjoindre la couleur des caractères `>{\color{`*`couleur`*`} \columncolor{`*`couleur`*`}}` ;
- la couleur de fond d'une cellule en mettant `\cellcolor{`*`couleur`*`}` en début de cellule.

Par exemple

```
\begin{tabular}{l l >{\columncolor{green}} l}
   1 & 2 & 3 \\
   a & b & c
\end{tabular}
```

```
\begin{tabular}{l l l}
   \rowcolor{cyan} 1 & 2 & 3 \\
   a & b & c
\end{tabular}
```

```
\begin{tabular}{l l >{\columncolor{green}} l}
   1 & 2 & 3 \\
   a & \cellcolor{gray} b & c
\end{tabular}
```

On a souvent recours à une alternance de couleurs d'une ligne à l'autre. On peut utiliser pour cela la commande `\rowcolors` (avec un *s*) :

```
\rowcolors{début}{couleur impaire}{couleur paire}
```

l'argument *début* est la ligne où commence cette alternance, ce qui permet d'épargner les lignes d'en-tête. Par exemple

```
 \rowcolors{2}{gray}{}
\begin{tabular}{l l >{\columncolor{green}} l}
   1 & 2  & 3 \\
   a & b  & c \\
   I & II & III
\end{tabular}
```

Notez que :

- une commande `\rowcolor` annule une commande `\columncolor` pour la ligne en cours ;
- une commande `\cellcolor` annule une commande `\columncolor` ou `\rowcolor(s)` pour la cellule en cours.

La commande `\arrayrulecolor{`*`couleur`*`}` permet de définir la couleur des filets (horizontaux et verticaux). Elle influe sur les filets tracés après cette commande ; elle peut donc être placée avant un tableau ou bien à l'intérieur de celui-ci.

> **Remarque :** dans toutes les commandes, `{`*`couleur`*`}` peut être remplacé par `[`*`modèle`*`]{`*`couleur`*`}`, cf. Programmation LaTeX/Options de mise en forme avancées > Modèles de couleur

Cas de `\multirow`

La fonction `\columncolor` colorie les cases une par une. Lorsque l'on utilise `\multirow`, on a donc :

1. La première case est colorée par `\columncolor`.
2. Le texte est écrit sur plusieurs lignes par `\multirow`.
3. Au passage à la ligne suivante, la deuxième case est colorée par `\columncolor`.

La deuxième ligne de texte est donc recouverte par la couleur de fond. Pour éviter ceci, il faut mettre le `\multirow` sur la dernière ligne et utiliser une valeur négative pour le nombre de lignes. Par exemple

```
\begin{tabular}{l >{\columncolor{lightgray}} l l}
   1 &   & 3 \\
   a &   & c \\
   I & \multirow{-3}{1em}{2 A II} & III
\end{tabular}
```

Tableaux longs

Si un tableau est susceptible d'être sur plusieurs pages, on utilise l'environnement `longtable`. Il s'utilise comme l'environnement `table` *et* comme l'environnement `tabular` :

```
\begin{longtable}{| p{0.2\linewidth} | p{0.2\linewidth} |}
\caption{Titre du tableau}
\hline
ligne 1 colonne 1 & ligne 1 colonne 2 \\
…
\end{longtable}
```

Il propose de plus des commandes supplémentaires :

- `\setlongtables` : cette commande est à mettre dans le préambule du document et permet d'utiliser tous les types de colonne (`<c`, `l`, …) ; si on l'omet, il faut indiquer la largeur des colonne (colonne de type `p`) ; comme l'affichage se fait sur plusieurs pages, le calcul des largeur nécessite plusieurs compilations ;
- `\endhead` : indique la fin de l'en-tête, remplace le dernier `\\` de cette partie ; la partie comprise entre le début de l'environnement et cette commande est répété à chaque changement de page ;
- `\endfoot` : indique la fin du pied de tableau, remplace le dernier `\\` de cette partie ; la partie comprise entre `\endhead` et cette commande est répété à chaque changement de page ;
- `\endfirsthead` et `\endfirstfoot` : s'utilise de la même manière, mais indique l'en-tête et le pied de tableau de la première page, s'ils sont différents de ceux des autres pages.

Notes de bas de page

En typographie, les notes de bas de page sont incompatibles avec les tableaux. De fait, si l'on utilise la commande `\footnote` dans un tableau, on a l'appel de note mais la note elle-même n'apparaît pas en bas de page.

Si l'on veut vraiment une note de bas de page et que celle-ci est unique, on peut mettre `\footnotemark` à l'appel de note, et placer `\footnotetext{`*`texte de la note`*`}` après l'environnement du tableau. Mais si l'on a plusieurs notes, il faut numéroter les notes à la main avec `\footnotetext[`*`n`*`]{`*`texte de la note`*`}`, ce qui pose le problème du suivi de la numérotation des notes.

Une solution plus satisfaisante consiste à mettre le tableau dans un environnement `minipage`, la note apparaît alors sous le tableau, mais pas en bas de la page.

Résumé

Nous en avons déjà assez dit au sujet des tableaux de base à mon avis. Après vous pouvez expérimenter afin d'en maîtriser le concept. Je dois admettre que la syntaxe d'un tableau en Latex peut sembler plutôt tordue, et ainsi voir des exemples plus sophistiqués pourrait vous embrouiller davantage. Mais heureusement, nous en avons assez vu pour que vous puissiez créer par vous-même n'importe quel tableau dont vous pouvez avoir besoin dans vos documents. Il est clair que Latex a beaucoup de ressources, ainsi attendez-vous à trouver des instructions plus avancées dans un proche avenir.

Images

À bien des égards, l'importation de vos images dans votre document en employant LaTeX est assez simple… à partir du moment où vos images sont dans le bon format ! Par conséquent, je crains que pour beaucoup de gens, le plus grand effort sera de convertir leurs fichiers graphiques.

Compilation avec LaTeX : Postscript encapsulé (EPS)

Fondamentalement, si vous souhaitez importer une quelconque image dans votre document en utilisant LaTeX, le format de fichier de l'image doit être EPS *(encapsulated PostScript)*. Le format EPS a été défini par Adobe pour rendre facile l'importation dans les documents, par les applications, des graphiques basés sur du PostScript. Puisqu'un fichier EPS déclare la taille de l'image, il devient aisé pour des systèmes comme LaTeX d'arranger le texte et les graphiques de la meilleure manière qu'il soit.

La plupart des logiciels de dessin dignes de ce nom, ont la possibilité d'enregistrer des images au format EPS (l'extension de ces fichiers est normalement `.eps`). Évidemment, les applications d'Adobe ont cette possibilité, puisqu'Adobe a développé les normes PS et EPS, toutefois il y a beaucoup d'autres solutions de rechange.

Conversion d'images sous Linux

Il existe tellement d'utilitaires sur Linux pour convertir un fichier graphique d'un format vers un autre que vous pouvez vous y perdre. Il y en a de nombreux qui sont spécialisés dans la conversion d'un format donné vers EPS. Citons `jpeg2ps` par exemple. Cependant, je parlerai d'un logiciel plus convivial et plus générique, appelé `ImageMagick` qui permet d'utiliser la commande `convert`.

`ImageMagick` supporte absolument tous les formats graphiques que vous pouvez mentionner, et sa beauté est qu'il vous laissera convertir un fichier de n'importe quel format vers n'importe quel format qu'il comprend. C'est un programme qui s'utilise en ligne de commande, mais très simple d'utilisation. Vous passez deux paramètres, le premier étant le nom de fichier de votre image courante, et le second est le nom de fichier que vous souhaitez donner à votre image convertie. Normalement, la seule différence est dans l'extension du fichier. Par défaut, `convert` déterminera les formats d'entrée et de sortie par les extensions des fichiers que vous lui fournissez.

Par exemple :

```
convert graphe.jpg graphe.eps
```

Cette commande charge le fichier au format JPEG, `graphe.jpg`, et le convertit en un fichier au format EPS portant le même nom. Naturellement, cela fonctionnerait également si votre image originale était au format BMP, GIF, PNG, etc.

Pour une approche plus graphique, essayez `The GIMP`, qui vous permet visuellement de charger et d'enregistrer dans de multiples formats d'image.

Conversion d'images sous Microsoft Windows

Il existe des applications de base sous Microsoft Windows qui peuvent faire ce travail de conversion, cependant, je ne retiendrai que quelques logiciels en vogue. Mon premier choix se porte évidemment sur `The GIMP` qui peut maintenant être installé sous Microsoft Windows, et mon deuxième choix `Paint Shop Pro` de Corel, qui est un logiciel commercial. Les deux logiciels gèrent un large éventail de formats graphiques, et conviendront parfaitement pour convertir les fichiers graphiques. À l'heure où est écrit cet article, la version de `The GIMP` est la 2.2.13 (nécessite `GTK+ 2` pour Microsoft Windows, version 2.8.18) et celle de `Paint Shop Pro` est la XI (11).

Même si `Paint Shop Pro` est plus largement utilisé, `The GIMP` est un logiciel libre de qualité professionnelle qui vous permettra de traiter les images sans débourser un euro ! Il est alors important que vous copiiez alors vos images à un endroit où Latex pourra les lire.

`CorelDraw X3` est également disponible pour les utilisateurs de Microsoft Windows qui peuvent réaliser des tâches semblables.

Naturellement, vous pouvez toujours lancer votre système Linux et utiliser l'utilitaire infiniment plus souple `convert`.

Il existe aussi des versions pour Microsoft Windows de la commande `convert`, comme par exemple celle proposée par la suite `ImageMagick`. Il faut cependant ne pas être effrayé par la ligne de commande sous Microsoft Windows…

Attention, le nom du fichier doit être écrit en un seul mot sinon il est possible que LaTeX ne trouve pas le fichier.

Compilation avec pdfLaTeX

Si vous compilez avec pdfLaTeX, vous pouvez utiliser quasiment tous les formats sauf… les formats PostScript (dont les images PSTricks). Si vous ne savez pas avec quel programme vous allez compiler, le mieux est d'avoir deux fichiers de formats différents pour la même image, présents dans le même répertoire et portant le même nom à l'exception de l'extension :

- pour des photographies, un fichier JPEG et un fichier EPS ;
- pour des dessins matriciels, un fichier PNG (ou GIF) et un fichier EPS ;
- pour des dessins vectoriels, un fichier PDF et un fichier EPS.

Il faut alors faire attention à *ne pas* indiquer l'extension dans la commande `\includegraphics` ; ainsi, le programme utilisé (LaTeX ou pdfLaTeX) choisira lui-même le « bon » fichier.

On peut aussi inclure des images au format EPS en appelant l'extension `eps2pdf` ou `epstopdf` , qui se charge de convertir les images. On peut utiliser PSTricks avec les extensions `pst-pdf` et `auto-pst-pdf`.

Extension `graphicx`

Je supposerai désormais que vous avez facilement converti toutes vos images dans le format EPS. Il est maintenant temps de voir les mécanismes permettant de les importer dans votre document LaTeX. Comme avec la plupart des choses en LaTeX, il y a plus d'un moyen d'éplucher une banane. Et cela reste vrai pour importer des fichiers au format EPS dans vos documents. Cependant, je m'intéresserai plus particulièrement à une extension appelée `graphicx`, qui effectue rapidement cette tâche, et avec peu d'effort.

Avant de pouvoir disposer des commandes suivantes pour travailler, vous devez placer `\usepackage{graphicx}` dans le préambule de votre document. La syntaxe pour utiliser *graphicx* est :

```
\includegraphics[attr1=val1, attr2=val2, ..., attrn=valn]{image}
```

Comme vous devriez maintenant le savoir si tout va bien, les paramètres entre crochets sont facultatifs, tandis que les accolades sont obligatoires. La variété d'attributs possibles qui peuvent être fournis est grande, et donc je ne parlerai que des plus utiles :

`width=xx`	Indiquez à la place de *xx* la largeur souhaitée de l'image importée.
`height=xx`	Indiquez à la place de *xx* la hauteur souhaitée de l'image importée.
NB: en ne fournissant que l'un ou l'autre des paramètres précédents, l'image changera de taille en respectant le même rapport de proportionnalité.	
`keepaspectratio`	ce paramètre peut prendre deux valeurs `true` ou `false`. *Lorsqu'il est placé à `true`, l'image sera changée de taille à la fois en hauteur et en largeur, mais l'image ne sera jamais aplatie de sorte que ni la largeur ni la hauteur ne seront dépassées.*
`scale=xx`	modifie la taille de l'image en suivant l'échelle donnée. Exemple 0.5 permet de réduire de moitié, ou 2 d'agrandir du double.
`angle=xx`	Cette option peut tourner l'image de *xx* degrés (sens opposé à celui de la rotation des aiguilles d'une montre)
`trim=g b d h`	Cette option aura pour effet de couper l'image importée de *g* à partir de la gauche, *b* à partir du bas, *d* à partir de la droite, et *h* à partir du haut ; où *g*, *b*, *d* et *h* sont des longueurs.
`clip`	Pour que l'option `trim` fonctionne `clip=true` doit être présente.

Pour utiliser plus d'une option en même temps, séparez les simplement avec des virgules.

Exemples

Note: pour adapter l'image à la taille de la page on peut utiliser la commande \textwidth:

```
\includegraphics[width=\textwidth]{poussin.eps}
```

Il est temps maintenant d'afficher des graphiques. Vous êtes, comme toujours, invité à regarder le fichier d'exemple à la fin du cours d'instruction. Regardez tout d'abord le fichier `.tex`, et ensuite le résultat `.pdf`. Voici quelques exemples tirés dudit fichier `.tex` :

```
\includegraphics{poussin.eps}
```

Cette commande importe l'image, cependant, elle est très grande (aussi je ne l'afficherai pas ici !). Aussi rapetissons-la :

```
\includegraphics[scale=0.5]{poussin.eps}
```

Elle a maintenant été réduite de moitié. Si vous souhaitez être plus spécifique et donner des longueurs réelles pour les dimensions de l'image, vous pouvez l'importer de cette façon :

```
\includegraphics[width=2.5cm]{poussin.eps}
```

Pour tourner l'image (je l'ai aussi rapetissée) :

```
\includegraphics[scale=0.5, angle=180]{poussin.eps}
```

Et finalement, voici un exemple montrant comment couper une image de façon à se focaliser sur une région particulière :

```
 \includegraphics[trim = 10mm 80mm 20mm 5mm, clip, width=3cm]{poussin.eps}
```

Remarquez la présence de `clip`, sans laquelle l'opération de découpage ne fonctionnerait pas.

Autres moyens de modifier l'image

Les commandes servant à modifier l'apparence du texte peuvent aussi servir à modifier l'image (voir *Mise en forme du texte (avancé)* > *Déformation du texte) :*

```
\scalebox{échelle}{%
   \includegraphics{image}%
}
```

pour dilater l'image ou la contracter d'un facteur *échelle* ;

```
\resizebox{largeur}{!}{%
   \includegraphics{image}%
}
```

pour fixer la largeur, ou bien

```
\resizebox{!}{hauteur}{%
   \includegraphics{image}%
}
```

Les paramètres *largeur* et *hauteur* sont un nombre accolé à une unité (cf. *Éléments de base > Espaces et changements de ligne*).

Position de l'image par rapport au texte

Pour LaTeX, une image est un objet graphique similaire aux lettres. Si l'on utilise simplement l'instruction `\includegraphics`, l'image sera placée au sein du paragraphe. La notion d'objet flottant permet de placer l'image à part, avec un titre (légende) et un numéro de figure auquel on peut faire référence — voir le chapitre suivant *Éléments flottants et figures*. Mais il existe d'autres moyens de placer la figure.

L'instruction `\marginpar` met un paragraphe — texte et/ou image — dans la marge. La syntaxe est

```
\marginpar{\includegraphics{monimage}\\ légende} Texte du paragraphe.
```

LaTeX choisit lui-même la marge (gauche ou droite). Pour placer l'image dans l'autre marge, on utilise `\reversemarginpar`.

L'extension `picins` fournit la commande `\parpic` qui permet d'avoir une image entourée de texte. La version de base de la commande est

```
\parpic{\includegraphics{monimage}} Texte du paragraphe.
```

On a alors une image située à gauche. On peut utiliser des options :

- `l`, `r` : place l'image respectivement à gauche (*left*, option par défaut) ou à droite *(right)* ;
- `f` : entoure l'image d'un filet ;
- `d` : entoure l'image de tirets *(dash)* ;
- `s` : cadre avec l'image avec une ombre *(shadow)* ;
- `o` : met des coins arrondis au cadre ;
- `x` : le cadre a un effet volumique.

Par exemple

```
\parpic[rs]{\includegraphics[scale=0.7]{monimage}} Texte du paragraphe.
```

Résumé

J'ai écrit tout ce que vous devez savoir pour importer vos images dans un document LaTeX. Comme je l'ai dit, la conversion proprement dite est probablement la partie la plus longue du processus entier, puisque la commande pour inclure l'image est très simple.

Il y a une rubrique importante manquant à ce cours d'instruction qui parlerait de la façon de faire devenir une image, *une figure*. Pour cela, vous aimeriez ajouter une légende, et ou y inclure une référence peut-être. Cependant, c'est délibéré, parce qu'il n'y a pas seulement les images qui peuvent être des figures. Par conséquent, ce sujet doit être traité dans un chapitre complet.

Éléments flottants et figures

Dans le précédent chapitre, l'importation des images a été abordée. Cependant, avoir juste une image coincée entre des paragraphes ne semble pas très professionnel. Pour les novices, nous voulons une manière d'ajouter des légendes, et pouvoir établir des références croisées. Ce dont nous avons besoin est une manière de définir des *figures*. Il serait également agréable si Latex pouvait appliquer des principes semblables à ceux utilisés quand il traite le texte, à l'arrangement des images, afin que tout paraisse au mieux. C'est ici que les éléments *flottants* rentrent en jeu.

Les flottants

Les éléments flottants se rapportent à tout ce qui, dans un document, ne peut pas être inséré dans une page. Ils se ramènent fondamentalement aux tables et aux figures. Ils exigent un traitement particulier, et le concept de *flottant* était la solution pour traiter de tels éléments, tout en maintenant la présentation de document aussi « belle » que possible.

Le problème le plus courant est la manifestation d'un manque de place sur le reste d'une page donnée pour placer une figure particulière. Pour surmonter cela, LaTeX fera nager celle-ci jusqu'à la page suivante, tout en remplissant la page courante avec le corps du texte. Comparez cela avec ce qui se produit sous *MS Word*, par exemple. Si une image est ajoutée mais est trop grande pour s'adapter à la page courante, il la placera sur la prochaine page, mais laissera un grand espace, au lieu de réarranger le texte aux alentours pour remplir l'espace. Cela exige beaucoup de manipulations délicates pour rectifier, alors que LaTeX prend en charge tout ceci automatiquement.

Les figures

Pour créer une figure, vous devez employer l'environnement *figure* (La Palice).

\begin{figure}[*paramètre indiquant le placement*]
... corps de la figure ...
\end{figure}

Dans la section précédente, j'expliquais comment des éléments flottants pouvaient être utilisés pour permettre au Latex de manipuler des figures, tout en maintenant la meilleure présentation. Cependant, il se peut que vous soyez en désaccord avec par exemple le positionnement des figures. Dans ce cas, il existe un paramètre du prescripteur de placement, qui vous donne un plus grand degré de contrôle de placement des éléments flottants.

Paramètre de position	Permission
`h`	Place le flottant *ici*, c'est-à-dire à l'endroit auquel il apparaît dans le texte source.
`t`	Position en *haut* de la page.
`b`	Position en *bas* de la page.
`p`	Place sur une *page* particulière réservée aux flottants.
`!`	Passe outre les paramètres internes que Latex utilise pour déterminer une position optimale des flottants.

Les *permissions de placement* vous permettent d'indiquer les options que vous souhaitez rendre disponibles au Latex. Ce sont simplement des possibilités, et Latex décidera quand il composera votre document lequel de vos prescripteurs fournis sera le meilleur.

Dans le cas ou vous avez beaucoup de flottants pour peu de texte et que LaTeX éloigne les flottants de leur contexte, la commande `\FloatBarrier` du paquet `placeins` permet de vider le tampon de flottants actuellement stockés, et donc de positionner tous les flottants déjà déclarés avant de poursuivre le document.

Les tableaux

Bien que les tableaux aient été déjà abordés, nous n'avions discuté que de la syntaxe interne. L'environnement `tabular` qui a été employé pour construire les tableaux n'est pas un élément flottant par défaut. Par conséquent, si vous souhaitez laisser flotter des tableaux vous devez envelopper l'environnement `tabular` avec un environnement `table`, comme ceci:

```
\begin{table}
  \begin{tabular}{...}
  ... données du tableau ...
  \end{tabular}
\end{table}
```

Vous avez peut-être l'impression que tout cela est un peu trop enveloppé, mais de telles distinctions sont nécessaires, parce que vous ne voulez pas forcément que tous les tableaux soient traités comme des flotteurs.

Les légendes

C'est toujours une habitude d'ajouter une légende à n'importe quelle figure ou tableau. Heureusement, c'est très simple avec LaTeX. Tout ce que vous avez besoin de faire est d'employer la commande `\caption{`*`texte`*`}` dans l'environnement de l'élément flottant. En raison de la façon dont Latex travaille avec les structures logiques, il maintiendra automatiquement la numérotation des figures, ainsi vous n'avez nul besoin de l'inclure dans le texte de la légende.

La position de la légende est traditionnellement sous l'élément flottant. Cependant, il vous appartient d'insérer la commande de légende après le

contenu même du flotteur (mais toujours dans l'environnement). Si vous le placez avant, alors la légende apparaîtra au-dessus du flotteur. Essayez l'exemple suivant pour montrer cet effet (voyez [tutorial6/legende.tex legende.tex] et [tutorial6/legende.pdf legende.pdf])

```
\begin{figure}
  \caption{Une image d'un toucan.}
  \centering
  \includegraphics{toucan.eps}
\end{figure}
```

```
\begin{figure}
  \centering
  \reflectbox{\includegraphics{toucan.eps}}
  \caption{Une image du même toucan regardant de l'autre côté!}
\end{figure}
```

Le package babel détermine le nom générique donné aux tableaux et aux figures dans la commande caption. Par défaut avec l'option \usepackage[french]{babel}, on obtient 'FIG -' et 'TAB -'. On peut modifier ces paramètres en ajoutant dans le préambule après le package babel les deux lignes suivantes :

```
\addto\captionsfrench{\def\figurename{Graphique}}
\addto\captionsfrench{\def\tablename{Tableau}}
```

ou

```
\addto\captionsfrench{\def\figurename{Graphique}\def\tablename{Tableau}}
```

Les étiquettes et les références croisées

C'est une bonne occasion d'introduire les *étiquettes*. Leur utilité en Latex est d'agir comme un marqueur qui peut être référencé à n'importe quel endroit de votre document. Il est très fréquent de devoir se rapporter à chacune des figures, dans le corps du texte. Cependant, vous n'aimeriez pas garder en mémoire leur numéro, ainsi vous pouvez employer une étiquette à la place, et laissez Latex la remplacer automatiquement par le numéro correct de la figure.

Pour ajouter une étiquette, vous devez inclure la commande suivante:

```
\label{marqueur}.
```

Pour ensuite référencer une étiquette : `\ref{`*marqueur*`}`.

Une autre solution serait d'utiliser une référence de page: `\pageref{`*marqueur*`}`.

```
\begin{figure}
  \centering
  \reflectbox{\includegraphics{toucan.eps}}
  \caption{Une image du même toucan regardant dans l'autre sens!}
  \label{toucan}
\end{figure}
```

```
La figure~\ref{toucan} montre une photographie de toucan.
```

Quand une étiquette est déclarée dans un environnement flottant, la commande `\ref` renvoie le numéro de figure/tableau correspondant (bien que, celui-ci puisse apparaître après la légende). Quand celle-ci est déclarée en dehors, la commande renvoie le numéro de section.

Le tilde (~) dans l'exemple ci-dessus est un symbole spécial en Latex. Il représente un espace insécable. Il est utile ici parce qu'il garde « figure » et le numéro quel qu'il soit auquel \ref se rapporte comme un tout, et ne les coupera pas sur une ligne ou une page lors de la production du document.

Enveloppement des figures

Bien que ce ne soit pas normalement utile dans le cadre de l'écriture d'un ouvrage scolaire, un auteur pourrait préférer que certains éléments flottants ne brisent pas le déroulement du texte, mais qu'au lieu de cela, permettent au texte de s'enrouler autour de ceux-ci. Évidemment, cet effet semble réalisable uniquement lorsque la figure en question est sensiblement plus étroite que la largeur du texte. Le paquet `wrapfig` a été conçu pour effectuer cette tâche.

Pour employer `wrapfig`, vous devez d'abord ajouter `\usepackage{wrapfig}` au préambule. Ceci vous donne alors accès à la commande `\begin{wrapfigure}{`*alignement*`}{`*largeur*`}`.

alignement peut être soit *l* pour la gauche, ou *r* pour la droite.

La *largeur* est évidemment la largeur de la figure. Un exemple:

```
\begin{wrapfigure}{r}{40mm}
  \centering
  \includegraphics{toucan.eps}
  \caption{Le toucan}
\end{wrapfigure}
```

Fichier:Enveloppement.png

Notez que l'environnement *wrapfigure* est en fait un élément non flottant. Cela signifie qu'il peut vous demander un certain soin afin que votre figure insérée ne passe pas au delà des coupures de page.

Sous-figures

Une extension utile est le paquet `subfig`, qui remplace le paquet `subfigure` depuis la distribution TeXLive 2010. Ce paquet donne au rédacteur la possibilité d'insérer des figures dans des figures. Les sous-figures ont leur propre légende, disposent aussi d'une légende globale facultative. Voici un exemple qui en illustre l'utilisation.

```
\usepackage{subfig}
...
\begin{figure}[htp]
  \centering
  \subfloat[Image d'origine]{\label{fig:edge-a}\includegraphics[scale=0.75]{toucan.eps}}
  \hspace{5pt}
  \subfloat[Après une détection des contours de Laplace]{\label{fig:contour-b}\includegraphics[scale=0.75]{laplace_tou
  \hspace{5pt}
  \subfloat[Après une détection des contours de Sobel]{\label{fig:contour-c}\includegraphics[scale=0.75]{sobel_toucan.
  \caption{Différents algorithmes de détection des contours}
  \label{fig:contour}
\end{figure}
```

Fichier:Sous-figure.png

Par rapport au paquet `subfigure`, il suffit simplement de modifier le préambule en conséquence, et de remplacer dans le texte source la commande `\subfigure` par `\subfloat`. Notez aussi que le paquet `caption` peut être requis par le paquet `subfig`.

Mise en forme du texte (avancé)

La mise en forme du texte est un terme plutôt large, mais dans le cadre de cette section elle se limitera à diverses techniques de composition d'un texte, de mise en page ou d'organisation de paragraphes.

La mise en forme tend à se rapporter à tout ce qui concerne l'aspect, et elle englobe des sujets comme les modèles de texte, les polices, la taille des caractères ; l'alignement de paragraphe, l'espacement entre les lignes, les indentations; les types particuliers de paragraphe ; les structures de liste ; les apostilles, les notes de marge, etc. Beaucoup de techniques de mise en forme d'un texte sont exigées pour différencier certains éléments du reste du texte. Il est souvent nécessaire d'ajouter de l'emphase à des mots ou des expressions. Une liste numérotée ou de description est généralement employée comme un moyen clair et concis de communiquer une question importante. Les apostilles sont utiles pour fournir l'information supplémentaire ou pour clarifier sans interrompre le déroulement principal du texte. Ainsi, pour toutes ces raisons, la mise en forme est très importante. Cependant, il est également très facile d'exagérer, et un document dont la rédaction abuse de ces techniques peut paraître encore moins lisible et esthétique qu'un autre qui n'en utilise pas du tout.

Mise en forme du texte

Guillemets

L'extension `csquotes` permet une grande souplesse dans la gestion des guillemets qui s'adaptent automatiquement au contexte. Cela est très utile car les guillemets français sont différents des guillemets anglais ou allemands.

L'option `babel=true` du package permet de faire en sorte que les guillemets correspondent à la langue définie dans l'extension `babel`.

La commande de base du package est `\enquote{texte à mettre entre guillemets}` et s'utilise par exemple ainsi :

```
\documentclass[a4paper,12pt,french]{article} % la langue du document est définie au niveau de la classe
\usepackage{babel} % sans option, babel choisit la langue en fonction de celle définie dans la classe du document
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}

\usepackage[babel=true]{csquotes} % csquotes va utiliser la langue définie dans babel

\begin{document}
Je cite : \enquote{citation}.
\end{document}
```

Il existe d'autres commandes de l'extension `csquotes` permettant notamment de changer de langue à l'intérieur d'un texte. Pour plus d'information, consulter la documentation sur la page de l'extension (http://www.ctan.org/tex-archive/macros/latex/contrib/csquotes) [archive] (en anglais).

Il est également possible d'utiliser les commandes `\og` et `\fg` du package babel : voir la documentation (http://daniel.flipo.free.fr/frenchb/frenchb2-doc.pdf) [archive].

Points et tirets

Une suite de trois points forme ce que l'on appelle des points de suspension, qui sont généralement employés pour indiquer qu'une partie du texte est omise. Le fait de juxtaposer simplement trois points ne donne pas exactement des points de suspension parce que l'espacement n'est pas correct.

Par conséquent, vous devriez employer `\ldots` en le plaçant à la fin d'un mot sans laisser d'espace pour obtenir le bon résultat.

LaTeX a quatre types spécifiques de tirets, chacun ayant une taille différente, et un emploi différent :

- la division, ou trait d'union « - » : `-` (moins), ou `\-` s'il est optionnel (voir *Options de mise en forme avancées > Césure)* ;
- le tiret demi-cadratin « – » : `--`, utilisé pour indiquer un intervalle, par exemple « cf. p. 5–7 », mais également pour les incises (d'une force « différente », à voir, par rapport aux parenthèses) ;
- le tiret cadratin « — » : `---`, est principalement utilisé pour le changement d'interlocuteur dans les dialogues ;
- le signe moins « - » : `$-$` ; l'usage veut l'emploi du demi-cadratin pour le différencier du trait d'union.

Le tiret cadratin rompt le gris optique, certains éditeurs préfèrent utiliser le tiret demi-cadratin pour l'incise.

Les accents

Il y a plusieurs moyens d'écrire des caractères accentués en LaTeX :

- La première façon est d'inclure dans le préambule, la commande `\usepackage[latin1]{inputenc}` qui permet d'utiliser les caractères ISO 8859-1, et il est alors possible de taper directement les caractères accentués à condition d'utiliser un éditeur de texte mettant à disposition ce jeu de caractères.

- On peut aussi utiliser le jeu de caractères UTF-8 qui est plus récent et qui a l'avantage de permettre de coder d'autres langages que le seul français. Malheureusement, il n'est pas encore compatible avec toutes les packages latex. La commande à inclure dans le préambule est alors `\usepackage[utf8]{inputenc}`.

- On peut enfin utiliser une des commandes particulières qui permettent de représenter les accents. Ces commandes sont assez intuitives, par exemple `\'` place un accent aigu sur le caractère qui suit, `\`` place un accent grave. De plus, elles ne nécessitent que l'encodage ASCII, universellement reconnu.

À noter que la conversion entre utf8 et iso8859-1 peut être faite par la plupart des éditeurs de texte récents ou alors par des utilitaires comme recode.

Symboles

LaTeX met plusieurs symboles à disposition. La plupart d'entre eux appartiennent au domaine mathématique, et les chapitres suivants expliquent comment y accéder. Pour les symboles plus courants des textes, les commandes suivantes sont disponibles :

Naturellement, ceux-ci ne sortent pas de l'ordinaire, et on peut largement préférer les symboles plus intéressants, de la police de *ZipfDingbats* de *Postcript* disponible grâce à `pifont`.

Maintenant, vous devez savoir que lorsque vous voulez employer une extension, vous devez ajouter une déclaration à votre préambule, dans ce cas : `\usepackage{pifont}`. Ensuite, la commande `\ding{`*nombre*`}`, imprimera le symbole indiqué. Vous pouvez consulter le tableau présentant tous les symboles possibles à *Table des caractères Unicode/U2700* (attention, \ding{} n'utilise pas les codes du lien précédents mais ceux-ci (http://willbenton.com/wb-images/pifont.pdf) [archive]).

Mise en évidence

Le meilleur moyen de mettre en évidence un mot ou une phrase dans un texte est d'employer la commande de `\emph{`*texte*`}`. Comme vous le voyez, il n'y a rien de plus simple.

Modèles de polices de caractères

Je n'approfondirai pas vraiment sur les polices de caractères dans ce paragraphe. Cette section ne traite pas de la façon d'obtenir un texte écrit dans une police de caractères *Verdana* de taille 12 points (12pt) ! Il existe trois familles principales de polices : *roman* (telle que *Times*), *sans serif* (exemple *Arial*) et *monospace* (exemple *Courier*).

Vous pouvez également indiquer des modèles tels que italique (italic) et gras (bold). Le tableau suivant présente les commandes qui permettent d'accéder aux modèles de police typiques :

<-- Fichier:Styles.png -->

Vous avez pu noter l'absence de soulignement. Cette fonctionnalité peut être ajoutée avec l'extension `ulem`. Placez `\usepackage{ulem}` dans votre préambule. Par défaut, la commande devient prioritaire par rapport à `\emph` et remplace le mode italique par le soulignement. Il est peu probable que vous souhaitiez obtenir cet effet, aussi afin d'éviter que l'extension `ulem` se substitue à `\emph`, il vaut mieux simplement appeler la commande de soulignement uniquement lorsque c'est nécessaire.

- Pour neutraliser `ulem`, ajoutez `\normalem` directement après le début de l'environnement.
- Pour souligner, employez `\uline{... }`.
- Pour souligner avec une ligne ondulée, employez `\uwave{... }`.
- Et pour barrer les caractères `\sout{... }`.

Enfin, il y a la question de la taille. Il est très facile de modifier la taille en utilisant les commandes du tableau.

Déformation du texte

Il est possible de déformer le texte, pour avoir des « effets spéciaux ».

Il est tout d'abord possible de décaler le texte vers le haut ou vers le bas :

```
\raisebox{décalage}{texte}
```

par exemple

```
du texte \raisebox{1ex}{plus haut} ou \raisebox{-1ex}{plus bas}.
```

Cette commande est différente de `\vspace` qui produit elle un blanc après la ligne en cours (un interligne plus grand).

La commande `\shortstack{… \\ …}` qui permet de mettre deux textes l'un au-dessus de l'autre.

Le décalage horizontal se fait avec `\hspace{`*longueur*`}`, qui peut servir à faire des grandes espaces, mais aussi à superposer du texte. Pour calculer la longueur que prend une portion de texte, il faut définir une longueur, avec `\newlength`, et calculer la longueur avec `\settowidth` ; le nom de la variable de longueur commence par une contre-oblique. Par exemple, si l'on veut décaler une portion de texte d'un dixième de cadratin

(`0.1em`) :

```
\newlength{\textlarg}
\newcommand{\dedoublement}[1]{%
   \settowidth{\textlarg}{#1}
   #1\hspace{-\textlarg}\hspace{0.1em}#1}

Exemple de \dedoublement{texte dédoublé}.
```

Dans la commande personnelle (macro), le texte est mis dans la variable `#1`, et sa largeur est mise dans `\textlarg`. Le texte est affiché, puis le « curseur » est ramené en arrière d'une valeur de `\textlarg` (donc au début du texte en question), puis avancé d'1/10 cadratin, puis le texte dans #1 est à nouveau affiché.

Avec un décalage plus faible (par exemple de `0.03em` ou `0.1ex`), on a un pseudo-gras. Avec un décalage plus important (par exemple de `0.15em` ou `0.3ex`), on peut avoir un effet de « texte ajouré ».

On peut utiliser un procédé similaire pour barrer du texte :

```
\newlength{\textlarg}
\newcommand{\barre}[1]{%
   \settowidth{\textlarg}{#1}
   #1\hspace{-\textlarg}\rule[0.5ex]{\textlarg}{0.5pt}}

Exemple de \barre{texte barré}.
```

La commande `\rule` produit ici un rectangle situé à une demie hauteur d'x de la ligne de base (`0.5ex`), d'un demi-point d'épaisseur (`0.5pt`) et ayant la longueur du mot (`\textlarg`). Voir aussi une autre solution ici.

L'extension `graphicx` fournit d'autres des commandes de déformation. La commande `\rotatebox{`*`angle`*`}{`*`texte`*`}` permet de faire tourner le texte, par exemple

```
Voici un \rotatebox{45}{texte} tourné de 45 degrés.
```

La commande `\scalebox{`*`facteur`*`}{`*`texte`*`}` permet de dilater ou contracter un texte. On peut aussi définir un facteur horizontal et vertical différent : `\scalebox{`*`facteur horizontal`*`}[`*`facteur vertical`*`]{`*`texte`*`}`, par exemple

```
Voici un \scalebox{2}[1]{texte} étiré en longueur.
```

La commande `\resizebox{`*`largeur`*`}{`*`hauteur`*`}{`*`texte`*`}` a un effet similaire, mais on indique la hauteur et la largeur du texte, par exemple

```
Voici un \resizebox{2cm}{1cm}{texte} dans une boîte de deux centimètres sur un.
```

On peut remplacer une des dimensions par un point d'exclamation `!` pour garder les proportions.

Enfin, la commande `\reflectbox{`*`texte`*`}` écrit le texte en miroir (en inversant la droite et la gauche). On peut inverser le haut et le bas en combinant avec une rotation, mais il faut penser à remonter le texte si l'on veut qu'il se trouve sur la ligne.

```
\newcommand{\miroirvert}[1]{%
   \raisebox{1ex}{\rotatebox{180}{\reflectbox{{#1}}}}}
```

Mise en forme d'un paragraphe

Le changement de la mise en forme d'un paragraphe n'est pas souvent nécessaire, comme dans l'écriture scolaire. Cependant, il est utile de savoir comment l'effectuer, et quelles en sont les applications dans la mise en forme du texte dans des éléments flottants, ou dans d'autres documents plus exotiques.

Alignement des paragraphes

Les paragraphes en LaTeX sont habituellement entièrement justifiés (c'est-à-dire, affleurant les côtés des deux marges de gauche et de droite). Si pour une raison quelconque, vous souhaitez changer la justification d'un paragraphe, alors LaTeX met à votre disposition trois environnements, ainsi que des commandes équivalentes.

Alignement	Environnement	Commande
Justifié à gauche	`flushleft`	`\raggedright`
Justifié à droite	`flushright`	`\raggedleft`

Centré	`center`	`\centering`

On notera que :

- les environnements `flushleft` et `flushright` font référence au côté de l'alignement, *flush* pouvant se traduire par « chasser vers » ;
- les commandes `\raggedright` et `\raggedleft` font référence au côté opposé ; *ragged* signifie « en lambeau », mais le terme en typographie française est « en drapeau » (drapeau droit pour l'alignement à gauche, drapeau gauche pour l'alignement à droite).

Tout le texte entre `\begin` et `\end` de l'environnement indiqué sera justifié convenablement. Les commandes citées servent dans d'autres environnements, comme par exemple les tableaux (cf Alignement du texte dans un tableau).

Alinéa (indentation des paragraphes)

En typographie, un alinéa est un retrait de la première ligne d'un paragraphe ; on parle aussi d'indentation. Ce retrait est en général assez petit. La taille du retrait est déterminée par un paramètre appelé le `\parindent`. Sa longueur par défaut est fixée par la classe du document que vous employez. Il est possible de l'imposer en utilisant la commande `\setlength`.

```
\setlength{\parindent}{longueur}
```

va fixer la taille du retrait à *longueur*.

Cependant, si vous décidez de remettre le retrait à zéro, alors vous allez certainement avoir besoin de laisser un espace vertical entre les paragraphes afin d'aérer le texte. L'espace entre les paragraphes est contenu dans `\parskip`, qui pourrait être modifié par un moyen semblable à celui ci-dessus. Cependant, ce paramètre est aussi utilisé dans d'autres environnements comme les listes, ce qui implique qu'en le modifiant, vous courrez le risque de faire ressembler certaines parties de votre document à un brouillon. (c'est-à-dire, à un texte bien plus désordonné que le type de modèle de paragraphe !) Il peut être préférable d'employer une classe de document spécialement conçue pour ce type d'indentation, telle que `artikel3.cls` (écrit par un hollandais, traduit en article3).

Espacement interligne

Il est rarement nécessaire d'utiliser un autre espace interligne que le simple espace. Mais pour ceux qui décideraient d'en changer, voici comment :

1. ajoutez `\usepackage{setspace}` au préambule de votre document ;
2. ceci fournit alors les environnements suivants prêts à être utilisés dans votre document :
   - `doublespace` - toutes les lignes sont doublement espacées ;
   - `onehalfspace` - espace interligne fixé à un espacement de un et demi ;
   - `singlespace` - espace interligne normal.

Paragraphes particuliers

Pour ceux d'entre vous qui ont lu une bonne partie de ce guide jusqu'à cette page, vous avez sans doute déjà rencontré plusieurs des formats de paragraphe suivants. Bien que nous en ayons déjà parlé auparavant, il n'est pas inutile de les réintroduire ici, dans le soucis d'être le plus complet possible.

Environnement `verbatim`

Cet environnement a été employé dans un exemple du chapitre précédent du guide. Toutes les données placées entre les commandes *begin* et *end* sont imprimées comme si elles avaient été tapées avec une machine à écrire. Tous les espaces et passages à la ligne sont reproduits comme ils se présentent, et le texte est affiché dans une police non proportionnelle appropriée. Cet environnement est idéal pour dactylographier un code source de programme, par exemple.

```
\begin{verbatim}
L'environnement verbatim
reproduit simplement tous
les caractères entrés,
y compris les espaces!
\end{verbatim}
```

Remarque : une fois dans l'environnement `verbatim`, la seule commande qui sera identifiée est `\end{verbatim}`. Toutes les autres seront écrites in extenso! Si vous désirez qu'une commande soient interprétée dans un tel environnement, alors vous pouvez employer l'extension `alltt` à la place.

Note : Il existe également un environnement `Verbatim` (il est nécessaire d'ajouter `\usepackage{fancyvrb}` à votre préambule) qui permet de définir un cadre et la taille du texte. Par exemple :

```
\begin{Verbatim}[frame=single,fontsize=\scriptsize]
1..2..3
\end{Verbatim}
```

```
\begin{alltt}
L'extension Verbatim permet
d'employer des commandes normales.
Par conséquent, il est par exemple possible
de \emph{mettre en évidence} des mots
dans cet environnement.
\end{alltt}
```

N'oubliez pas d'ajouter `\usepackage{alltt}` à votre préambule, avant d'utiliser cette commande.

Environnement `listing`

Il s'agit également d'une extension de l'environnement *verbatim*. La fonctionnalité supplémentaire qu'elle fournit est celle de pouvoir ajouter des numéros de ligne de chaque côté du texte. Pour cela on emploie la commande : `\begin{listing}[pas]{première ligne}`. Le paramètre obligatoire *première ligne* indique à quelle ligne la numérotation débutera. Et le paramètre facultatif *pas* représente le pas entre chaque numéro de ligne (le pas par défaut vaut 1, et cela signifie les lignes seront numérotées normalement).

Pour employer cet environnement, ajoutez `\usepackage{moreverb}` au préambule de votre document.

Environnements `quote` et `quotation`

Il existe deux environnements permettant d'inclure des citations dans vos documents, avec une différence subtile entre ceux-ci. L'environnement `quote` est conçu pour de courtes citations, ou des séries de petites citations, séparées par des interlignes. D'un autre côté `quotation`, sert à incorporer au texte de plus longues citations, qui tiennent sur plus d'un paragraphe. Toutes les citations sont placées en retrait de l'une ou l'autre des deux marges, et vous devrez les entourer vous-même par des guillemets si vous le désirez.

Voyez le [tutoriel7/paras.tex paras.tex] et [tutoriel7/paras.pdf paras.pdf] pour des exemples d'utilisation de ces environnements.

Environnement `Abstract`

Dans certaines publications scolaires, un résumé est toujours placé au début. Il est généralement composé différemment du reste du texte, mis en retrait par rapport à l'un ou l'autre côté, et écrit avec une police légèrement plus petite. Il s'agit du moins de la présentation par défaut d'un résumé en LaTeX.

Voyez [latextutorial2.html tutorial 2], ou [tutorial7/paras.tex paras.tex] et [tutorial7/paras.pdf paras.pdf] pour trouver des exemples d'utilisation d'*abstract*.

Environnement `Verse`

Cet environnement est destiné à afficher de la poésie. Apparemment, peu de personnes utilisent cet environnement dans leur document, cependant, il ne coûte rien d'expliquer brièvement son utilisation. Comme tout environnement, il a besoin des commandes `\begin` et `\end`. Entre ces marques de début et de fin, un nouvelle strophe est créée en laissant une ligne blanche, et pour passer à la ligne dans une strophe on utilise la commande de passage à la ligne `\\`. Si une phrase prend plus d'une ligne dans la page, alors toutes les lignes suivantes seront mises en retrait par rapport à la première, jusqu'à ce que `\\` soit éventuellement rencontrée.

Voyez à ce sujet les [tutoriel7/paras.tex paras.tex] et [tutoriel7/paras.pdf paras.pdf] pour des exemples d'utilisation de cet environnement.

Structures de liste

Les listes apparaissent souvent dans les ouvrages, et plus particulièrement dans les ouvrages scolaires, car leur but est souvent de présenter l'information de façon claire et concise. Les structures de liste en LaTeX sont simplement des environnements qui se présentent sous trois formes différentes : `itemize`, `enumerate` et `description`.

Toutes les listes suivent le format de base :

```
\begin{type_de_liste}

  \item Premièrement
  \item Deuxièmement
  \item Troisièmement etc.

\end{type_de_liste}
```

`Itemize`

Cet environnement permet d'afficher des listes à puces.

```
\begin{itemize}
```

```
  \item Premier élément
  \item Deuxième élément
  \item Troisième élément etc.
\end{itemize}
```

Enumerate

Cet environnement est conçu pour représenter des listes numérotées, dans lesquelles chaque élément est numéroté successivement.

```
\begin{enumerate}
  \item Premier élément
  \item Deuxième élément
  \item Troisième élément etc.
\end{enumerate}
```

Description

L'environnement de description est légèrement différent. Vous pouvez donner une étiquette à un élément de la liste en la passant comme paramètre facultatif (bien que facultatif, il paraîtrait curieux de ne pas l'inclure!). Cet environnement est idéal pour une série de définitions, comme on en trouve dans un glossaire par exemple.

```
\begin{description}
  \item[Premier] Le premier élément
  \item[Deuxième] Le deuxième élément
  \item[Troisième] Le troisième élément
\end{description}
```

Listes imbriquées

LaTeX vous permet heureusement d'insérer un environnement de liste dans une autre liste existante (jusqu'à une profondeur de quatre). Il suffit simplement de créer un environnement de liste approprié à l'endroit désiré d'une liste donnée. LaTeX se chargera de la disposition des éléments, et d'une quelconque numérotation pour vous.

```
\begin{enumerate}
  \item Le premier élément
  \begin{enumerate}
    \item Premier élément de la liste imbriquée
    \item Deuxième élément de la liste imbriquée
  \end{enumerate}
  \item Le second élément
  \item Le troisième élément
\end{enumerate}
```

Listes personnalisées

Adapter des objets à ses besoins en LaTeX n'est pas toujours à la portée des débutants. Bien que ce ne soit pas forcément difficile intrinsèquement, les débutants, déjà accablés par une très grande quantité de commandes et d'environnements, risquent de se perdre en passant à des rubriques plus avancées.

Cependant, ce chapitre traite de la mise en forme d'un texte et je vais donner de brèves indications sur la façon de personnaliser les listes. Vous êtes libre de passer ces paragraphes pour le moment.

Listes numérotées personnalisées

Ce que souvent les personnes veulent changer dans les listes numérotées sont les compteurs. Par conséquent, pour mieux comprendre, nous avons besoin d'introduire brièvement les *compteurs* de LaTeX. À tout objet que LaTeX numérote automatiquement, comme les en-têtes de section, les figures, et les listes, est associé un compteur qui contrôle la numérotation. De plus chaque compteur possède un format par défaut qui dicte à LaTeX la façon dont il doit être imprimé. De tels formats sont modifiés en utilisant des commandes internes de LaTeX :

Commande	Exemple
`\arabic`	1, 2, 3 ...
`\alph`	a, b, c ...
`\Alph`	A, B, C ...

`\roman` i, ii, iii ...

`\Roman` I, II, III ...

`\fnsymbol` destinés à la numérotation des apostilles (voir ci-dessous), mais imprime une suite de symboles.

Il existe quatre compteurs différents qui sont associés aux listes à puces, chacun représentant les quatre niveaux possibles d'imbrication, et ils s'appellent : `enumi`, `enumii`, `enumiii`, `enumiv`. Chaque compteur contient plusieurs bits de données fournissant différentes informations. Pour obtenir l'élément numéroté, employez simplement la commande `\the` suivie immédiatement (c'est-à-dire sans aucun espace) du nom du compteur, par exemple `\theenumi`. Cette information est souvent désignée sous le nom de *représentation du compteur*.

Maintenant, laissons la plupart des technicités de côté. Pour effectuer des changements sur la mise en forme d'un niveau donné :

```
\renewcommand{\représentation}{\commande_de_mise_en_forme{compteur}}.
```

Évidemment, la version générique n'est pas vraiment claire, aussi une paire d'exemples clarifiera :

```
% Redéfinition du premier niveau
\renewcommand{\theenumi}{\Roman{enumi}}
\renewcommand{\labelenumi}{\theenumi}
```

```
% Redéfinition du deuxième niveau
\renewcommand{\theenumii}{\Alph{enumii}}
\renewcommand{\labelenumii}{\theenumii}
```

La méthode utilisée ci-dessus change d'abord explicitement le format d'impression employé par le compteur. Cependant, l'élément qui contrôle l'étiquette doit être mis à jour pour refléter le changement, et cet ajustement est effectué par la deuxième ligne. Une autre manière d'obtenir le même résultat est :

```
\renewcommand{\labelenumi}{\Roman{enumi}}
```

Cela redéfinit simplement l'aspect de l'étiquette, ce qui suppose que vous n'avez pas l'intention d'établir des renvois à un article spécifique de la liste, auquel cas la référence serait imprimée dans l'ancien format. Ce problème n'apparaît pas dans le premier exemple.

Listes à puces personnalisées

Les listes à puces ne sont pas aussi complexes que les listes numérotées puisqu'elles n'utilisent pas de compteur. Ainsi, pour personnaliser de telles listes, vous pouvez juste changer les puces (étiquettes). Les puces sont accessibles via les commandes `\labelitemi`, `\labelitemii`, `\labelitemiii`, et `\labelitemiv` pour les quatre niveaux respectifs.

```
\renewcommand{\labelitemi}{\textgreater}
```

L'exemple ci-dessus imposerait aux puces du premier niveau d'être représentées par le symbole « > » strictement supérieur. Bien sûr, les symboles de texte en LaTeX ne sont pas très impressionnants. Pourquoi ne pas utiliser les symboles de la polices *ZapfDingbat*, déjà utilisée dans la section *Symboles*?

Si on utilise babel en français, les puces ne sont plus accessibles via `\labelitemi`, mais via `\FrenchLabelItem`. Par exemple, pour changer les puces de tous les niveaux :

```
\renewcommand{\FrenchLabelItem}{\textbullet}
```

Pour changer les puces niveau par niveau, il faut alors utiliser les commandes `\Frlabelitemi`, `\Frlabelitemii`, `\Frlabelitemiii` et `\Frlabelitemiv` au lieu des commandes de type `\labelitemi`, par exemple pour le premier niveau :

```
\renewcommand{\Frlabelitemi}{\textgreater}
```

Titres

Il est possible de modifier l'apparence des titre de parties, chapitre, sections, …

Numérotation

La numérotation des parties fonctionne de manière similaire à celle des listes numérotées (voir *Listes numérotées personnalisées*) : la mise en forme du numérotage est définie par une commande, et il faut redéfinir cette commande pour changer cette mise en forme. La numéro de la partie est dans un compteur

Mise en forme de la numérotation des parties

Niveau de partie	Commande de mise en forme du niveau	compteur
partie	`\part`	`part`
chapitre	`\chapter`	`chapter`
section	`\section`	`section`
sous-section	`\subsection`	`subsection`
sous-sous-section	`\subsubsection`	`subsubsection`
paragraphe	`\paragraph`	`paragraph`
sous-paragraphe	`\subparagraph`	`subparagraph`

Pour redéfinir la mise en forme de la numérotation, on utilise alors `\renewcommand{`*`commande de niveau`*`}{`*`mise en forme`*`}`. Par exemple, pour mettre le numéro de section en chiffres romains capitales, on met dans le préambule :

```
\renewcommand{\thesection}{\Roman{section}}
```

cela va cependant poser problème dans la table des matières, puisque les chiffres romains sont plus longs que les chiffres arabes. Il faut donc augmenter l'espace entre les chiffres et le titre dans la table des matières :

```
\usepackage{tocloft}
\addtolength{\cftsecnumwidth}{1em}
\addtolength{\cftsubsecnumwidth}{1em}
```

ou bien

```
\usepackage{tocloft}
\newlength{\malong}
\setlength{\malong}{1em}
\addtolength{\cftsecnumwidth}{\malong}
\addtolength{\cftsubsecnumwidth}{\malong}
```

Texte

Pour modifier la mise en forme du texte des titres, on peut utiliser l'extension `sectsty` ou l'extension `fncychap`.

Notes de pied de page

Les notes de pied de page sont très utiles pour fournir au lecteur des informations complémentaires. Elles ne sont pas en principe essentielles, et peuvent ainsi être placées au pied de la page, ce qui a pour effet de laisser le corps principal du texte concis.

Il est facile de placer des notes au pied de la page. La commande dont vous avez besoin est : `\footnote{`*`texte`*`}`. Ne laissez pas d'espace entre la commande et le mot près duquel vous souhaitez que le marqueur d'apostille apparaisse, sinon LaTeX tiendra compte de cet espace et donnera un résultat différent de celui attendu.

```
La création d'une note de pied de page est d'une grande simplicité.\footnote{Voici un exemple de note de pied de page.}
```

LaTeX prendra évidemment soin de placer l'apostille au bas de la page. Chaque apostille est numérotée consécutivement. Ce processus, comme vous l'aviez certainement deviné est automatiquement fait à votre place.

Il est possible d'adapter à votre guise l'impression des notes de pied de page. Par défaut, elles sont numérotés consécutivement en utilisant les chiffres arabes. Cependant, sans trop s'enfoncer dans les mécanismes internes de LaTeX pour le moment, il est possible de les changer en utilisant la commande suivante (qui a besoin d'être placée au début du document, ou tout au moins avant la première commande d'apostille dans le texte).

`\renewcommand{\thefootnote}{\arabic{footnote}}`	Numérotation avec les chiffres arabes exemple 1, 2, 3...
`\renewcommand{\thefootnote}{\roman{footnote}}`	Numérotation avec les chiffres romains (en minuscule), exemple i, ii, iii...
`\renewcommand{\thefootnote}{\Roman{footnote}}`	Numérotation avec les chiffres romains (en majuscule), exemple I, II, III...
`\renewcommand{\thefootnote}{\alph{footnote}}`	avec les lettres de l'alphabet (en minuscule), exemple a, b, c...
`\renewcommand{\thefootnote}{\Alph{footnote}}`	avec les lettres de l'alphabet (en majuscule), exemple A, B, C...
`\renewcommand{\thefootnote}{\fnsymbol{footnote}}`	Une suite de neuf symboles (essayez et regardez le résultat!)

Notes de marge

C'est une commande peu utilisée, malgré sa simplicité d'utilisation. `\marginpar{`*`texte dans la marge`*`}` placera *texte dans la marge* dans la marge de gauche. Pour changer le côté par défaut, il suffit d'utiliser dans le préambule la commande `\reversemarginpar` qui obligera LaTeX à placer les notes dans la marge du côté opposé.

Liens hypertextes

Le package `hyperref` crée des liens sur la table des figures, ainsi que la table des matières, et permet d'inclure des liens hypertexte dans le document :

- `\href{http://fr.wikipedia.org}` inclura par exemple un lien vers *Wikipédia*.
- `\href{http://fr.wikipedia.org}{Wikipédia}` inclura un lien avec l'étiquette *Wikipédia*.
- `\hyperlink{label}{texte du lien}` permettra d'atteindre la cible identifiée par `\hypertarget{label}{texte de la cible}`. Les textes du lien et de la cible peuvent être vides.

Vous pouvez également régler certains paramètres d'un document PDF, grâce à la commande `\hypersetup{}`, placée par exemple dans le préambule du document :

```
\hypersetup{
   backref=true,                          % Permet d'ajouter des liens dans
   pagebackref=true,                      % les bibliographies
   hyperindex=true,                       % Ajoute des liens dans les index.
   colorlinks=true,                       % Colorise les liens.
   breaklinks=true,                       % Permet le retour à la ligne dans les liens trop longs.
   urlcolor= blue,                        % Couleur des hyperliens.
   linkcolor= blue,                       % Couleur des liens internes.
   bookmarks=true,                        % Créé des signets pour Acrobat.
   bookmarksopen=true,                    % Si les signets Acrobat sont créés,
                                          % les afficher complètement.
   pdftitle={Mon document au format TeX},  % Titre du document.
                                          % Informations apparaissant dans
   pdfauthor={PoluX},                     % dans les informations du document
   pdfsubject={Projet wikiBooks}          % sous Acrobat.
}
```

Résumé

Ce chapitre a dû vous paraître vraiment long. Mais la mise en forme d'un texte demande quelques approfondissements si vous désirez obtenir une présentation de votre document parfaite.

LaTeX est tellement flexible et paramétrable que j'ai seulement effleuré le sujet. En réalité il existe de très nombreux contrôles qui vont vous permettre d'obtenir la présentation la plus soignée possible et surtout celle que vous souhaitez.

Mais un des buts de LaTeX est justement de vous éviter le plus possible de devoir vous occuper vous-même de la présentation, ainsi ne vous en souciez pas trop pour l'instant.

Choix de la police

LaTeX utilise par défaut les fontes Computer Modern (CM) ou Extended Computer Modern (EC), ainsi que Latin Modern (LM) (voir *Premier exemple > Amélioration du code source*).

Il est alors possible de changer de fonte avec `\textsf{`*`texte`*`}` ou `\texttt{`*`texte`*`}`, et pour une fonte de changer de police avec les commandes `\emph{`*`texte`*`}`, `\textbf{`*`texte`*`}`, `\textsc{`*`texte`*`}` (voir *Mise en forme du texte > Choix de la police*).

Le mode mathématiques permet d'utiliser d'autres fontes (caligraphique avec `\mathcal{`*`texte`*`}`, alphabet grec, et avec l'extension `amsmath`, alphabet gothique avec `\mathfrak{`*`texte`*`}`), mais il n'est pas du tout adapté aux textes.

Mais on peut utiliser d'autres fontes que les fontes (Extended) Computer Modern et Latin Modern. Toutefois, toutes ne contiennent peut être pas tous les symboles, il faut donc les choisir avec soin selon ses besoins.

Notons que LaTeX 2ε utilise un système de gestion des polices appelé NFSS *(new font selection scheme)*, qui permet de les manipuler simplement (par exemple d'avoir du gras et italique en même temps). Il importe donc de ne retenir, parmi les solutions que l'on peut trouver sur le Net, que celles qui utilisent le système NFSS.

Police Metafont et police PostScript

Selon certaines sources, si l'on génère un fichier PDF, il vaut mieux utiliser une police PostScript. Notons toutefois que l'extension `lmodern` a réglé les problèmes de rendu des fontes Computer Modern dans un PDF.

Toutes les fontes PostScript ne conviennent pas. Si vous achetez une fonte PostScript propriétaires, commerciale, elle n'est pas utilisable directement avec LateX, et sa transposition sort du cadre de ce wikilivre.

Les polices Metafont sont composée de deux fichiers : un fichier portant l'extension `.tfm` *(TeX font metric)*, et un fichier portant l'extension `.pk` ou `.gf`. Vous pouvez avoir la liste des polices en recherchant les fichiers `.tfm` sur le disque dur.

Les fontes PostScript sont constituées de deux fichiers : un fichier portant l'extension `.afm` *(Adobe font metric)* et un portant l'extension `.pfb`. Pour une utilisation en LaTeX, le fichier `.afm` doit être converti en fichier `.tfm`…

Polices fournies par des extensions

Certaines extensions permettent de changer de police, par exemple :

- l'extension `fourier` [16] (http://www.ctan.org/tex-archive/fonts/fourier-GUT/) permet d'utiliser la fonte Utopia ;
- l'extension `mathptmx` permet d'utiliser la fonte Times ;
- l'extension `mathpazo` [17] (http://www.ctan.org/tex-archive/fonts/mathpazo/) permet d'utiliser la fonte Palatino ;
- l'extension `aeguill` permet d'utiliser la fonte CM en PostScript ;
- les extensions `concrete` [18] (http://www.ctan.org/tex-archive/fonts/concrete/) et `euler` pour les fontes de même nom (Euler est une fonte mathématiques, que l'on appelle avec les arguments optionnels `\usepackage[mathcal,mathbf]{euler}`) ; pour l'ecriture cursive *(script)* en fonte Euler (mode mathématiques), on utilise `\mathscr` ;
- l'extension `oldgerm` fournit la commande `\textgoth{`*`texte en gothique`*`}` ; l'extension `yfonts` fournit la commande `{\frakfamily `*`texte en gothique`*`}` ;
- l'extension `frcursive` [19] (http://www.ctan.org/tex-archive/fonts/frcursive/) [20] (http://www.pps.jussieu.fr/~beffara/soft/frcursive/) fournit l'environnement `cursive`, qui imite une écriture manuelle.

Utilisation des polices Metafont

Metafont est un langage de description des polices créé spécifiquement pour TeX (et donc LaTeX) par le créateur de TeX, Donald E. Knuth.

Les distributions de LateX disposent de plusieurs fontes Metafont ; les polices CM, EC et LM en font partie.

Le nom des fichiers est en général en trois parties :

1. le nom de la fonte ;
2. le type de caractère : « m » pour la graisse moyenne (non gras), « b » pour gras *(bold)*, « c » pour condensé, « n » pour la forme normale (non italique), « it » pour la forme italique, « sc » pour petites capitales, « sy » pour les mathématiques (symboles) … ;
3. le corps de la police, en points.

Par exemple, le fichier `cmb10.tfm` correspond à la police Computer Modern gras de corps 10.

Utiliser une police Metafont

Pour utiliser une police Metafont, vous pouvez taper

```
\fontfamily{fonte}\fontseries{graisse}\fontshape{forme}\selectfont
```

par exemple, pour passer à la police Dunhill (cmdh) de graisse moyenne (m), taper

```
\fontfamily{cmdh}\fontseries{m}\selectfont
```

Ce changement est « définitif » (jusqu'au prochain `\selectfont`), sauf si vous prenez la précaution d'inclure ceci dans un bloc `{…}`.

Vous pouvez utiliser dans le préambule

```
\DeclareTextFontCommand{\nom_personnel}{\fontfamily{fonte}\selectfont}
```

la commande `\nom_personnel` s'utilise alors comme les `\emph`, `\textbf` et consorts. Cela permet de séparer le fond de la forme…

Par exemple

```
\DeclareTextFontCommand{\helvetica}{\fontfamily{phv}\selectfont}
…
\helvetica{Texte en Helvetica}
```

Les commandes habituelles — `\emph`, `\textbf`, … — sont alors utilisables à l'intérieur (cette commande est compatible avec le système NFSS).

Si l'on veut utiliser cette police par défaut, il faut alors mettre dans le préambule

```
\renewcommand{\familydefault}{fonte}
```

Installer une police metafont

Il existe des polices Metafont à télécharger. Elles se présentent sous la forme d'un fichier `.mf`.

C'est le programme `mf` qui génère les fichiers exploitables (`.tfm` et .pk ou `.gf`) à partir du `.mf` :

```
mf nom_du_fichier
```

Il peut y avoir quelques problèmes de rendu avec certains programmes : `mf` considère par défaut que l'on utilise une imprimante d'un certain type, ce qui n'est pas forcément le cas des autres programmes. On peut essayer

```
mf \mode=ljfour; input nom_du_fichier
```

pour régler le problème.

Il faut ensuite mettre les fichiers dans un répertoire dédié par l'installation, en général du type `texmf.local/fonts/source/`, ou sous MacOS X `~/Library/texmf`. Dans un système Unix, on a la liste des répertoires possibles en tapant `echo $TEXMF`, il faut alors en choisir un contenant le mot « local ».

On l'utilise après comme ci-dessus.

Voir aussi

- Les fontes sous LaTeX pour les nuls (et les autres) (http://zoonek.free.fr/LaTeX/Fontes/fontes.html) [archive]
- Exemples de fontes (http://benoit.rivet.free.fr/tex/tex_polices_exemples.htm) [archive]
- Fonts with TeX and XeTeX on MacOS X (http://tug.org/mactex/fonts/) [archive]
- Langues exotiques (http://www.tuteurs.ens.fr/logiciels/latex/langues.html) [archive] (grec, cyrilique, japonais, chinois), ENS
- La police French Cursive (http://iml.univ-mrs.fr/~beffara/soft/frcursive/) [archive]

Mise en page

La mise en page est gérée directement par la classe de document. Lorsque votre travail doit être soumis à un éditeur pour être publié, la mise en page ne vous appartient pas et ce sont les éditeurs qui vous imposeront la présentation. Cependant, pour vos documents personnels, il y a quelques éléments dans la page que vous pouvez souhaiter modifier comme les marges, l'orientation du texte, les colonnes, etc. Le but de ce chapitre est de vous montrer comment configurer les paramètres pour la mise en page.

Les dimensions de la page

Une page en LaTeX est définie par une myriade de paramètres internes. Chaque paramètre correspond à la longueur d'un élément de la page, par exemple, `\paperheight` est la taille physique de la page. L'extension `layout` permet de visualiser dynamiquement la mise en page d'un document (bien que le résultat soit légèrement réduit), et fournit également les valeurs de plusieurs dimensions, qui peuvent être très significatives. Regardez [tutorial8/playout.pdf mise en page (pdf)] avant de continuer.

Il faut prendre garde au fait que la taille d'une page par défaut est celle des documents étasuniens, c'est-à-dire celle de *US letter* et non celle du format *A4*. Les pages sont en longueur plus courtes de 3/4 pouce, et légèrement plus larges de 1/4 pouce, comparé au format *A4* (qui est la norme en France et au Royaume-Uni). Ce n'est pas trop gênant parce que la plupart des imprimantes imprimeront tout de même la page en entier. Cependant, il est possible d'indiquer d'autres tailles, avec l'option *a4paper* de la classe de document.

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

L'exemple ci-dessus illustre comment passer le paramètre facultatif à `\documentclass`, qui modifiera alors les dimensions de la page en conséquence. Les classes standards de document font partie de LaTeX et ont été conçues de façon à être assez génériques, ce qui explique pourquoi vous avez la possibilité d'indiquer la taille de la page. D'autres classes peuvent avoir plusieurs options, ou au contraire aucune. Normalement, les classes sont documentées pour vous aider à les utiliser.

De plus, il existe plusieurs extensions prévues pour permettre la modification des dimensions d'une page, en changeant les valeurs par défaut de tous les paramètres de la classe de document. Une extension comme `a4` est plutôt spécialisée dans un seul type de page. L'une des extensions les plus souples qui s'adapte aux besoins de mise en page est `geometry`. Elle sera utilisée un certain nombre de fois dans ce guide parce qu'elle dispose de beaucoup de possibilités. Quoi qu'il en soit, pour fixer la taille d'une page, ajoutez simplement la ligne suivante à votre préambule:

```
\usepackage[a4paper]{geometry}
```

On peut également indiquer le ou les paramètres avec l'instruction `\geometry` fournie par l'extension :

```
\usepackage{geometry}
\geometry{a4paper}
```

L'option `a4paper` correspond simplement à l'une des nombreuses tailles de page prédéfinies intégrées. Les autres sont par exemple `a0paper`, `a1paper`, ..., `6paper`, `b0paper`, `b1paper`, ..., `b6paper`, `letterpaper`, `legalpaper`, `executivepaper`.

Orientation de la page

Quand vous entendez parler de changement d'orientation de la page, cela évoque habituellement un passage au format à l'italienne, ou paysage, le mode portrait étant celui par défaut. Nous présentons deux façons légèrement différentes de changer l'orientation de la page. La première est celle qui consiste à placer tout votre document en mode paysage dès le début. Il existe diverses extensions pour réaliser ce changement, mais le plus simple est l'extension `geometry`. Il suffit pour l'utiliser, d'inclure cette extension avec l'option *landscape* (paysage).

```
\usepackage[landscape]{geometry}
```

Si de plus vous avez l'intention d'employer l'extension `geometry` avec une option pour fixer la taille de la page, n'exécutez surtout pas la commande *\usepackage* deux fois, mais regroupez simplement toutes les options ensemble, en les séparant par des virgules :

```
\usepackage[a4paper,landscape]{geometry}
```

ou bien

```
\usepackage{geometry}
\geometry{a4paper,landscape}
```

La deuxième méthode permet d'écrire un document en mode portrait et d'afficher une certaine partie du contenu de la page en mode paysage pour une raison d'occupation d'espace ou esthétique. Cette partie pourrait comporter par exemple un très grand diagramme ou une table qui seraient mieux affichés dans l'autre sens. Mais vous voulez que vos en-têtes et titres de pied de page apparaissent au même endroit que dans les autres pages. L'extension `lscape` est spécialisée dans cette tâche. Elle fournit un environnement `landscape`, et tout ce qui se trouve dans cet

environnement est tourné dans un sens. Les dimensions de la page ne sont pas modifiées. Cette possibilité peut être appliquée aux livres et aux rapports, comme aux publications scolaires.

Dans le cas d'un tableau ou d'une figure flottante, l'extension `rotating` fournit les environnements `sidewaystable` et `sidewaysfigure` qui remplace les environnements `table` et `figure`.

Modification des marges

Les lecteurs qui utilisent régulièrement un traitement de texte classique, se demandent probablement pourquoi il y a tellement d'espace blanc entourant un texte produit avec LaTeX. Il y a une bonne raison à cela, et elle est directement en rapport avec la lisibilité. Allez lire quelques livres, et sélectionnez des lignes au hasard. Comptez le nombre de caractères par ligne. Je parie que la moyenne est d'environ 66 caractères. Les études ont prouvé qu'il est plus facile de lire le texte quand il y a entre 60 et 70 caractères par ligne, et il semblerait que 66 soit le nombre optimal. Par conséquent, les marges de page sont positionnées afin d'assurer la meilleure lisibilité possible. Par ailleurs, la disposition des blancs suit le procédé dit des « blancs tournants » (cf. *Rédaction technique* > *Lisibilité* > *Livre*). On ajoute souvent un blanc du côté de la marge intérieure au cas où le document aurait la prétention d'être relié, cela doit laisser l'espace nécessaire pour lier les pages.

Ce sont en fait les traitements de texte standard qui définissent des marges trop petites !

Cependant les marges produites par les classes standards — `article.cls`, `report.cls`, `books.cls` — peuvent apparaître trop larges aux yeux d'un Européen utilisant du papier A4. Une possibilité est d'utiliser des classes alternatives comme celles proposées par le projet KOMA-Script[1] — `scrartcl.cls`, `scrreprt.cls`, `scrbook.cls`. Une autre possibilité est de modifier les paramètres de la mise en page à l'aide de l'extension `geometry`. Il y a quatre paramètres prévus pour les marges de page qui peuvent être modifiées au moyen de cette extension: *top*, *bottom*, *left*, *right* (haut, bas, gauche, droit).

```
\usepackage[top=longueur, bottom=longueur, left=longueur , right=longueur]{geometry}
```

ou bien

```
\usepackage{geometry}
\geometry{top=longueur, bottom=longueur, left=longueur , right=longueur}
```

Remplacez simplement *longueur* par la longueur désirée (par exemple. `3cm`) pour chaque paramètre que vous voulez changer.

Styles de page

Personnalisation avec l'extension `fancyhdr`

L'appel à l'extension `fancyhdr` est sans doute la manière la plus commode de façonner vos en-têtes et pieds de page. Cette extension est très souple d'utilisation, et je vais simplement vous donner un avant goût de ce que vous pouvez faire avec. Vous pourrez trouver un guide plus complet, en consultant la documentation (http://www.ctan.org/tex-archive/macros/latex/contrib/fancyhdr/fancyhdr.pdf) [archive] écrite par l'auteur de l'extension.

Pour commencer, ajoutez les lignes suivantes à votre préambule:

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

L'en-tête et le pied de page comportent chacun trois parties correspondant à des positions horizontales différentes (à gauche, au centre ou à droite). Pour fixer leurs valeurs, les commandes suivantes sont disponibles:

`\lhead[`*lh-even*`]{`*lh-odd*`}` `\lfoot[`*lf-even*`]{`*lf-odd*`}`
`\chead[`*ch-even*`]{`*ch-odd*`}` `\cfoot[`*cf-even*`]{`*cf-odd*`}`
`\rhead[`*rh-even*`]{`*rh-odd*`}` `\rfoot[`*rf-even*`]{`*rf-odd*`}`

L'effet produit par chacune de ces commandes se devine assez facilement: si une commande contient le mot *head*, alors elle va affecter l'entête de la page etc., et évidemment, *l*, *c* et *r* signifient gauche (*left*), centre (*centre*) et droite (*right*) respectivement. Les documents peuvent être recto ou recto-verso. Les articles sont par défaut recto, et les livres sont recto-verso. Les documents recto-verso différencient les pages de gauche (impaires) et de droite (paires), tandis que les documents recto ne font pas cette différence. Un exemple:

```
\fancyhead{}
\fancyfoot{}

\lhead{Andrew Roberts}
\rhead{\today}
\rfoot{\thepage}
```

Page de n à m

Certaines personnes aiment numéroter les pages du document entier en indiquant éventuellement le nombre total de pages ; mais Latex ne permet d'accéder qu'au numéro de la page courante et par défaut vous ne pourrez que placer au bas de la page ce numéro. Cependant, vous pouvez inclure l'extension `lastpage` pour déterminer le nombre total de pages, et l'employer de la manière suivante:

```
\usepackage{lastpage}
...
\cfoot{\thepage\ sur \pageref{LastPage}}
```

Remarquez les lettres majuscules de *LastPage*. Ainsi, en plaçant une barre oblique inversée après \thepage pour laisser un espacement approprié entre le numéro de page et le « sur ». Et rappelez-vous que lorsque vous utilisez des références, vous devez exécuter *latex* une nouvelle fois pour résoudre les références croisées.

Pages multicolonnes

Il est courant de rencontrer des articles possédant deux colonnes de texte. Heureusement, les éditeurs fournissent souvent les classe de document qui permettent de formater le texte de cette façon sans demander de travail à l'auteur. Même si une telle classe de document n'est pas fournie, il est aisé de formater le texte en deux colonnes. Il suffit de passer l'argument *twocolumn* à la classe de document courante. Pour un article par exemple cela donne: `\documentclass[twocolumn]{article}`

Bien que cette solution fonctionne dans 9 cas sur 10, elle comporte certaines limitations qui sont éliminées par l'extension `multicol`. De plus cette extension possède les avantages suivants :

- Supporte jusqu'à 10 colonnes,
- Fournit un environnement *multicol* qui permet d'utiliser différents nombre de colonnes dans un même document,
- Cet Environnement peut être inclus dans d'autres environnements comme `figure`,
- Les colonnes produites par cette extension sont *équilibrées*, ce qui permet d'obtenir des colonnes de taille similaires pour la dernière page,
- La séparation verticale entre colonnes peut être paramétrée.

Les éléments flottant ne sont pas complètement supportés par cette extension. Elle traite cependant bien les éléments qui s'étendent sur toutes les colonnes (c'est-à-dire du type `\begin{figure*}`).

Le code suivant montre comment créer une partie de texte sur deux colonnes:

```
\begin{multicols}{2}
  Du texte très intéressant qui doit
  être mis sur deux colonnes.
  \ldots
\end{multicols}
```

Le paramètre `\columnseprule` contrôle la largeur du trait séparant les deux colonnes. Sa valeur par défaut est zéro, ce qui a pour résultat de ne pas tracer de ligne entre les colonnes. Pour obtenir une ligne de un point de large entre les colonnes il faut insérer le code suivant avant l'environnement `multicol`:

```
\setlength{\columnseprule}{1pt}
```

Le paramètre `\columnsep` contrôle l'espace entre les colonnes. Pour espacer les colonnes de deux centimètres, il faut insérer le code suivant:

```
\setlength{\columnsep}{2cm}
```

Mise en forme des pages d'un manuel

Sommaire

Ce guide est relativement court, et cela est dû en grande partie au fait que la philosophie de Latex est de laisser le rédacteur se concentrer sur le contenu, et de prendre en charge (en utilisant éventuellement des classes appropriées développées par des typographes) l'organisation de la présentation. La prochaine étape pour mieux maîtriser l'organisation de la présentation d'une page est de commencer à concevoir votre propre classe. Malheureusement, ce n'est pas une tâche aisée, et ce fardeau est souvent laissé aux professionnels !

Notes et références

1. Le projet KOMA-script a été créé à l'origine (1994) pour la typographie allemande. Les classes proposent des options intéressantes et sont de fait utilisées maintenant par des utilisateurs d'autres langues.

Mathématiques

L'une des motivations majeures à la définition du langage *TeX* et au développement du logiciel correspondant par Donald Knuth était de faciliter la composition de formules mathématiques tout en garantissant une qualité professionnelle à l'impression. Sa réussite dans la réalisation de ces deux buts a été la cause du succès de *TeX*, et plus tard de *LaTeX*, auprès de la communauté scientifique. *TeX* est en effet le langage le plus utilisé pour la composition et la restitution de formules mathématiques complexes à l'écran.

Cependant, le langage reste difficile à acquérir en raison du grand nombre de symboles mathématiques existants. Le but de cet article est de permettre au lecteur d'acquérir les bases de LaTeX.

Les environnements mathématiques

LaTeX doit savoir à l'avance si le texte à traiter contient ou non des éléments mathématiques car il compose les notations mathématiques différemment du texte normal : la police change (bien que par défaut elle ressemble à celle du texte) ainsi que l'espacement (« `ab` » correspond à « a multiplié par b » et non à une syllabe d'un mot), …

Des environnements spéciaux ont donc été définis à cette fin. Ils peuvent être répartis en deux catégories selon la façon dont ils sont présentés et peuvent apparaître

- dans le texte, au sein d'un paragraphe : les formules mathématiques sont affichées dans une ligne, c'est-à-dire dans le corps du texte là où elles sont écrites, par exemple dans la phrase « nous en déduisons que $a+a=2a$, après simplification. » ;
- de façon isolée (mode mathématique isolé, *displayed* en anglais), en dehors d'un paragraphe : les formules affichées sont séparées du texte principal, elles sont centrées sur la page ou la colonne (ou du moins, s'il y a plusieurs lignes, la ligne la plus grande est centrée).

Les équations hors paragraphe, isolées, peuvent être numérotées ; outre le fait que LaTeX numérote de manière automatique, si l'on place une étiquette `\label{…}` dans l'équation, la référence à cette étiquette (`\ref{…}`) donne le numéro de l'équation.

Il existe donc plusieurs environnements mathématiques, qui ont une forme « abrégée » :

Type	Environnement	Abréviation LaTeX	Abréviation Tex
dans le texte	`\begin{math}…\end{math}`	`\(…\)`	`$…$`
mode hors paragraphe	`\begin{displaymath}…\end{displaymath}`	`\[…\]`	$$ ... $$
mode hors paragraphe numéroté	`\begin{equation}…\end{equation}`	*aucune*	

Note

On voit dans de nombreux documents l'utilisation de `$$…$$`. Cette forme est à proscrire : elle donne un espacement incorrect et ne gère pas les options de classes [21] (http://www.tuteurs.ens.fr/logiciels/latex/maths.html). Par exemple, cette forme ne supporte pas les équations de nombre du côté gauche (classe `fleqn`).

Si l'on utilise l'extension `amsmath`, les modes hors paragraphe numéroté et non-numéroté sont équivalents, excepté la présence ou l'absence de numérotation. Si l'on n'utilise pas `amsmath`, les deux modes ne gèrent pas l'espacement vertical de la même manière.

Donnons tout de suite quelques exemples.

Le code `$\sqrt{\frac{x^2}{3}}$` va insérer la formule mathématique $\sqrt{\frac{x^2}{3}}$ dans un texte.

Autre exemple : `\[\sqrt{\frac{x^2}{3}}\]`. Cet exemple va afficher la même formule, sauf qu'elle va être centrée sur une nouvelle ligne.

```
\begin{document}
  Voici la fonction $\sin x$,
  la fonction \[\cos x=\sum_{k=0}^{\infty}\frac{(-1)^k}{(2k)!}x^{2k}\]
  et la fonction $\tan (x)$\ldots
\end{document}
```

va donner le résultat suivant :

Voici la fonction $\sin x$, la fonction

$$\cos x = \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k)!} x^{2k}$$

et la fonction tangente $\tan(x)$…

Autre différence avec la balise `\[…\]` par rapport à la balise `$…$` :

- `$\sum_{k=0}^2$` va donner

$\sum_{k=0}^2$

- alors que `\[\sum_{k=0}^2\]` va donner

$$\sum_{k=0}^2$$

Que faut-il écrire dans les balises `$…$` et `\[…\]` ?

Regardons premièrement, les puissances (exposants) et les indices des variables. Ceci est très simple, vous allez voir. Exemple, pour insérer x_i^2, il suffit d'entrer le code LaTeX suivant : `$x_i^2$`. Le symbole « ^ » met le caractère suivant en exposant, alors que le caractère « _ » (tiret de soulignement) le met en indice. Pour mettre un groupe de caractères ou toute une fonction en exposant (ou indice) il suffit d'utiliser les balises de groupes `{` et `}` (accolades). Par exemple, `$x^{x^2}$` va afficher x^{x^2}

Comment encadrer une équation ?

Pour encadrer une équation, on utilise l'extension `empheq`. Elle fournit l'environnement `empheq`, qui peut prendre plusieurs paramètres, en particulier :

- le type d'encadrement (paramètre optionel), avec `[box=…]` ; par défaut (paramètre absent), il n'y a pas d'encadrement ;
- le type d'environnement de base : `{equation}` pour une équation hors texte numérotée, `{equation*}` pour une équation non numérotée.

Par exemple :

```
\usepackage{empheq}
…
\begin{empheq}[box=\fbox]{equation*}
   \| \vec{\Cmath}_{3/1} \| = \unit{8\,670}{\newton}\text{.}
\end{empheq}
```

Les symboles

Les mathématiques utilisent beaucoup de symboles. S'il y a bien une difficulté en Latex c'est de se rappeler la façon dont les produire. Il y a bien sûr un ensemble de symboles directement accessibles à partir du clavier :

`+ - = ! / ( ) [ ] < > | ' :`

En dehors de ceux-ci, différentes commandes doivent être appelées pour afficher les symboles désirés. Et il existe beaucoup de lettres grecques, de symboles relationnels et ensemblistes, de flèches, d'opérateurs binaires, etc. Un très grand nombre de symboles à apprendre, et en fait, ils surchargeraient ce cours d'instruction si j'essayais de les énumérer tous. Par conséquent, vous pouvez consulter un document complet de référence à [tutorial9/symbols.pdf symbols.pdf]. Nous verrons naturellement certains de ces symboles utilisés dans tout ce cours.

Multiplication

Plusieurs symboles sont habituellement utilisés pour représenter les produits.

Il est possible de placer un point entre les deux nombres comme dans l'exemple : `$x \cdot y\ $` ce qui donne $x \cdot y$. Notez bien que la multiplication utilise le point centré (`\cdot`) et non pas un point normal.

La multiplication peut aussi être représentée par le symbole « `\times` » qui signifie « fois ». Par exemple, `$x \times y$` donne le résultat suivant : $x \times y$.

Fractions

Pour créer une fraction, vous devez utiliser la commande `\frac{`*num\acute{e}rateur*`}{`*d\acute{e}nominateur*`}`. Ce qui donne : $\frac{numérateur}{dénominateur}$

(Pour ceux qui ont besoin d'un moyen mnémotechnique, il suffit de retenir de *haut* en *bas* tout simplement.) Pour pouvez également imbriquer des fractions dans des fractions, comme le montrent les exemples ci-dessous :

`\frac{x+y}{y-z}` $\frac{x+y}{y-z}$

`\frac{\frac{1}{x}+\frac{1}{y}}{y-z}` $\frac{\frac{1}{x}+\frac{1}{y}}{y-z}$

Remarque : TeX fournissait une variante, la commande `\over`. Cependant, son usage est relativement malheureux car il place *tout* ce qui le précède au numérateur et *tout* ce qui le suit au dénominateur. Les cas où il peut être utile sont les fractions simples dans le texte, du type :

`1+x \over 2` $1+x \over 2$

Puissances et indices

Les puissances et les indices sont les équivalents mathématiques des indices supérieurs et inférieurs dans un texte normal. Le caractère chapeau « ^ » est utilisé pour élever quelque chose, et celui de soulignement « _ » pour abaisser. Voyons un exemple qui montre comment les utiliser :

Puissance	`x^n`	x^n
	`x^{2n}`	x^{2n}
Indice	`n_i`	n_i
	`n_{ij}`	n_{ij}

Remarque : s'il y a plus d'un caractère à élever (ou à abaisser) alors vous devez les grouper en utilisant les accolades (`{…}`).

De plus, si vous avez besoin d'assigner en même temps, à la même entité, une puissance et un indice, alors cela peut être réalisé de la manière suivante :

```
x^{2i}_{3j}
```

ou

```
x_{3j}^{2i}
```

l'ordre n'ayant pas d'importance. Les deux écritures donneront x_{3j}^{2i}

Racines

Typiquement, et presque tout le temps, vous êtes amenés à utiliser la racine carrée, qui est obtenue facilement en utilisant la commande suivante : `\sqrt{x}`. Bien que `\sqrt` vienne de *square root* qui signifie « racine carrée », la commande peut être généralisée pour produire une racine de n'importe quelle ordre :

`\sqrt[n]{param}` $\sqrt[n]{x}$

LaTeX s'assure automatiquement que la taille du symbole racine s'ajuste à la taille du contenu.

Si le paramètre *n* est omis, LaTeX produira une racine carrée.

Parenthèses et autres délimiteurs

Les différents délimiteurs

L'utilisation des parenthèses devient très rapidement importante même dans les relations les plus insignifiantes. Sans les parenthèses, les formules peuvent devenir ambiguës. En outre, les types particuliers de structures mathématiques, telles que les matrices, se fondent typiquement sur des parenthèses pour les englober.

Vous pouvez vous rappeler que vous avez déjà les symboles du `()` et `[]` à votre disposition, qui devraient être adaptés aux besoins de la plupart des rédacteurs.

Il existe en fait beaucoup plus de symboles possibles qui peuvent être employés à la place des crochets, mais sont quelque peu rares. Consultez par exemple le tableau 5 de la référence de symboles ([tutorial9/symbols.pdf symbols.pdf]) pour les autres caractères. En voici quelques-uns :

- `()` (parenthèses simples)
- `[]` (crochets)
- `|` (barre verticale)
- `\|` (barre verticale double)
- `\{ \}` (accolade ouvrante et fermante).
- `.` (délimiteur nul, voir *Remarque* ci-dessous)

Grands délimiteurs

Considérons l'exemple suivant :

```
\[ (\frac{x^2}{y^3}) \]
```

$$(\frac{x^2}{y^3})$$

```
\[ \left(\frac{x^2}{y^3}\right) \]
```

$$\left(\frac{x^2}{y^3}\right)$$

Le premier exemple montre ce qui se produirait si vous employiez les caractères standards de parenthèses. Comme vous pouvez le voir, ils conviendraient à une simple équation qui tiendrait sur sur une seule ligne (par exemple, (3 + 2) x (10-3) = 35) mais pas à des équations ayant une plus grande taille verticale, comme celles comportant des fractions. Le deuxième exemple illustre la manière dont Latex fait face à ce problème.

Les commandes `\left...` et `\right...` permettent d'adapter automatiquement la taille des parenthèses. Vous devez faire précéder les deux délimiteurs de ces commandes. Les points après la commande devront être remplacés par un des caractères selon le modèle de parenthèse voulu.

Une autre solution est d'utiliser le paquet *nath*, qui fournit des délimiteurs s'adaptant automatiquement à la taille.

Attention, en Latex, { et } sont interprétés : écrire `{x+1}` produit le résultat $x+1$: pas d'accolades. Pour les caractères { et }, il faut utiliser `\{` et `\}`, par exemple `\{x+1\}` produit le résultat $\{x+1\}$.

Remarque : pour utiliser un délimiteur d'un seul côté (par exemple mettre une accolade devant deux ou trois équations empilées), il faut utiliser le *délimiteur nul* qu'est le point : `\left. \right.`. Par exemple :

```
\left\{
  \begin{array}{rcr}
    x+2y & = & -1 \\
    -x+4y & = & 0 \\
  \end{array}
\right.
```

$$\left\{\begin{array}{rcr} x+2y & = & -1 \\ -x+4y & = & 0 \end{array}\right.$$

(voir la section Matrices pour plus d'explications sur l'environnement `array`)

Espacement horizontal

LaTeX modifie l'espacement horizontal selon le contexte. La plupart des délimiteurs sont reconnus comme tels : l'espacement à l'intérieur d'une paire ouvrante-fermante est différent de l'espacement à l'extérieur. Cependant, cela n'est pas toujours le cas :

- LaTeX ne sait pas déterminer si les délimiteurs `|` et `\|` sont ouvrant ou fermants (puisqu'ils sont identiques à gauche et à droite) ;
- l'adjonction des commandes `\left` et `\right` modifie cet espacement.

Pour éviter cela, on peut utiliser les commandes `\mathopen{}` en ouverture et `\mathclose{}` en fermeture. Par exemple :

```
\documentclass[10pt]{article}

\usepackage{amsmath}

\begin{document}

\begin{align}
  y &= \sin | x | \\
  y &= \sin \mathopen{|} x \mathclose{|} \\
  x &= \sin \left (\frac{1}{2} \right ) \\
  x &= \sin \mathopen{}\left ( \frac{1}{2} \right )\mathclose{}
\end{align}

\end{document}
```

$$y = \sin |x| \quad (1)$$
$$y = \sin|x| \quad (2)$$
$$x = \sin\left(\frac{1}{2}\right) \quad (3)$$
$$x = \sin\left(\frac{1}{2}\right) \quad (4)$$

Utilisation de `\mathopen` et `\mathclose`.

À l'inverse, on peut lui demander de considérer un délimiteur comme un caractère « normal » avec la commande `\mathord{}`.

Si l'on a des problèmes d'espacement à l'intérieur d'une formule, on peut encapsuler la formule dans un `\mathinner{…}` : elle est alors considérée comme étant « à l'intérieur » *(inner)* de délimiteurs (même si ceux-ci sont absents). Par exemple, pour le produit de fractions :

```
\documentclass[10pt]{article}

\usepackage{amsmath}
```

```
\begin{document}

\begin{align}
  x &= \frac{1}{2} \frac{3}{4} \\
  x &= \mathinner{\frac{1}{2}} \mathinner{\frac{3}{4}}
\end{align}

\end{document}
```

Utilisation de `\mathinner`.

Les fonctions mathématiques

Code LaTeX	Aperçu
\sin	$\sin$
\cos	$\cos$
\tan	$\tan$
\arcsin	$\arcsin$
\arctan	$\arctan$
\arccos	$\arccos$
\exp	$\exp$

On peut créer soi-même des fonctions mathématiques avec la commande `\DeclareMathOperator{`*`nom de la commande`*`}{`*`texte`*`}` du paquet amsmath : le code

```
\DeclareMathOperator{\argsinh}{argsinh}
```

définit une commande `\argsinh` qui affiche le texte "argsinh" de la même façon que les commandes `\sin`, ...

Sommes

Le signe de sommation :

```
\[\exp(x)=\sum_{k=0}^{\infty}\frac{x^k}{k!}\]
```

donne

$$\exp(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!}$$

(`\infty` donne le signe infini).

Remarque : ceci est valable pour le mode "mathématiques affichées" (hors-texte et centrées). Dans le mode "mathématiques en ligne" (dans le texte, ouvert par `\(` ou `$` et fermé par `\)` ou `$`), les bornes n'apparaissent pas sous et sur le symbole, mais respectivement en indice et en exposant. Pour obtenir le bon résultat, il faut utiliser la commande `\sum\limits_{`*`borne inf`*`}^{`*`borne sup`*`}`. Le mot-clé `\limits` indique que l'indice doit être en réalité placé *dessous* et l'exposant *dessus*. Ce mot-clé fonctionne avec tous les grands opérateurs (`\sum`, `\prod` etc.) décrits ci-dessous.

Produits

Le signe de produit :

```
$n!=\prod_{k=1}^n k$
```

donne

$$n! = \prod_{k=1}^{n} k.$$

Intégrales

L'intégrale :

```
$\int_a^b f(x) \, \mathrm dx$
```

donne

$$\int_a^b f(x)\,\mathrm{d}x$$

(le code « `\,` » insère une espace fine et le code « `\mathrm` » permet d'obtenir un « d » romain).

Matrices

Latex n'a pas une commande particulière pour créer une matrice. Il dispose à la place d'un environnement un peu plus général, appelé tableau ou `array`. L'environnement `array` est fondamentalement équivalent à l'environnement `tabular` (voyez le chapitre sur les tableaux pour vous remémorer la syntaxe d'une table). Les rangées sont paramétrables, et peuvent être employées dans beaucoup de cas de figure, mais nous nous concentrerons sur les matrices. Vous pouvez employer les tableaux pour placer et aligner vos données comme vous voulez, et ensuite l'entourer avec des parenthèses gauches et droites appropriées pour obtenir votre matrice. Pour une matrice (2, 2) simple :

```
\[ \left(
  \begin{array}{ c c }
     1 & 2 \\
     3 & 4
  \end{array} \right)
\]
```

$$\left(\begin{array}{ c c }
1 & 2 \\
3 & 4
\end{array}\right)$$

Voyons encore un autre exemple :

```
$\begin{array}{l|cr}1 & 2 & 3\\ \hline 4 & 5 & 6\\ 7 & 8 & 9\\\end{array}$
```

va donner

1	2	3
4	5	6
7	8	9

Après la déclaration de l'environnement, il faut insérer le type et le nombre de colonnes. Ici, `{l|cr}` :

- `l` : les éléments de la colonne sont alignés à gauche (comme `align=left` en HTML) ;
- `c` : les éléments de la colonne sont centré sur la colonne (comme `align=center` en HTML) ;
- `r` : les éléments de la colonne sont alignés à droite (comme `align=right` en HTML) ;
- `|` : insert un trait vertical entre deux colonnes.

À l'intérieur de l'environnement, on peut trouver les codes suivants :

- `\hline` : insert une ligne horizontale sur toute la largeur du tableau ;
- `&` : passe à la colonne suivante (comme `<td>` en HTML) ;
- `\\` : passe à la ligne suivante (comme `</tr>` en HTML).

Pour une documentation plus approfondie sur les tableaux, allez voir le chapitre sur les tableaux.

Si vous incluez beaucoup de mathématiques dans vos documents, alors probablement vous vous apercevrez rapidement que vous souhaitez avoir un peu plus de contrôle sur la présentation. Certaines possibilités rendent l'écriture des formules plus complexes — mais qui a dit que la composition des mathématiques était facile ?

Les caractères grecs

Code LaTeX	Aperçu
\alpha	α
\beta	β
\gamma	γ
\delta	δ
\eta	η
\theta	θ
\vartheta	ϑ

Code LaTeX	Aperçu
\pi	π
\sigma	σ
\tau	τ
\omega	ω
\phi	ϕ
\varphi	φ

Code LaTeX	Aperçu
\Pi	Π
\Sigma	Σ
\Tau	T
\Omega	Ω
\Phi	Φ

Je pense que vous avez compris le principe. Je ne vais donc pas tous les énumérer (pour l'instant).

Ajouter du texte dans les formules

Je doute que ce soit tous les jours que vous aurez besoin d'inclure du texte dans une formule. Cependant, il arrive parfois que ce soit nécessaire. Un simple collage d'un texte directement dans l'environnement mathématique ne vous donnera pas les résultats que vous attendez. Par exemple :

```
\[
```

```
  50 pommes \times 100 pommes = beaucoup de pommes^2
\]
```

$$50pommes \times 100pommes = beaucoupdepommes^2$$

ou

```
\[
  5 \acute{e}l\acute{e}phants + 4 \acute{e}l\acute{e}phants=  9 \acute{e}l\acute{e}phants
\]
```

$$5éléphants + 4éléphants = 9éléphants$$

Il y a deux problèmes visibles. Premièrement, il n'y a aucun espace entre les nombres et le texte, ni d'espaces entre les mots entre eux. Deuxièmement, les mots n'apparaissent pas correctement; les lettres sont davantage espacées par rapport à la normale. Cela vient du fait que ce sont simplement des objets façonnés du mode mathématique qui n'attend pas ces mots. Tous les espaces que vous saisissez en mode mathématique sont ignorés et Latex espace les éléments selon ses propres règles. Il suppose que tous les caractères représentent des noms de variable. Pour montrer que chaque symbole est individuel, ils ne sont pas placés les uns à côté des autres aussi étroitement que dans du texte normal.

Il y a un certain nombre de manières selon lesquelles le texte peut être ajouté correctement. La manière typique est d'envelopper le texte avec la commande `\mbox{...}`. Cette commande n'a pas été présentée avant, cependant son travail est fondamentalement de créer une boîte juste assez large pour contenir le texte considéré. Le texte placé dans cette boîte ne peut pas être cassé sur plusieurs lignes. Voyons ce qui se produit quand le code de la formule ci-dessus est adapté :

$$50\text{pommes} \times 100\text{pommes} = \text{beaucoup de pommes}^2$$

Le texte a un meilleur aspect. Cependant, il n'y a aucun espace entre les nombres et les mots. Malheureusement, vous êtes contraints d'ajouter vous-mêmes ces derniers. Il y a de nombreuses façons d'ajouter des espaces entre les objets mathématiques, mais pour la simplicité, je trouve plus facile, au moins dans ce cas, d'insérer directement un caractère d'espace dans la boîte `\mbox` elle-même (juste avant le texte).

```
\[
  50 \mbox{ pommes} \times 100 \mbox{ pommes} =
  \mbox{beaucoup de pommes}^2
\]
```

$$50 \text{ pommes} \times 100 \text{ pommes} = \text{beaucoup de pommes}^2$$

Il est préférable d'utiliser la commande `\text` du package amsmath ; `\text` adapte la largeur du texte à la largeur du texte environnant :

- `$\underbrace{x + \cdots + x}_{10 \mbox{ fois}}$` va donner

$$\underbrace{x + \cdots + x}_{10 \text{ fois}}$$

- alors que `$\underbrace{x + \cdots + x}_{10 \text{ fois}}$` va donner

$$\underbrace{x + \cdots + x}_{10 \text{ fois}}$$

La mise en caractère romain dans les formules

Il est une convention d'écrire les variables en italique (cela se fait par défaut, ex. : `$x^2$`). On peut mettre du texte en romain. Par exemple,

```
$p_\mathrm{ext}$
```

va donner : p_{ext}.

Les fonctions s'écrivent également en romain ; les fonctions les plus courantes possèdent leur abréviation.

Mise en forme du texte

Employer `\mbox` permet de résoudre correctement le problème de base. Cependant, il y a une alternative qui offre un peu plus de flexibilité. Vous pouvez relire le chapitre sur la mise en forme du texte, l'introduction aux commandes de modification de forme de police, comme `\textrm`, `\textit`, `\textbf`, etc. Ces commandes écrivent leur paramètre en utilisant une certaine forme de police, et par exemple `\textbf{texte en caractères gras}` donne **texte en caractères gras**. Ces commandes sont également valables dans un environnement mathématique et permettent d'inclure du texte. L'avantage supplémentaire ici est que vous pouvez avoir un meilleur contrôle de la forme de la police de caractères, plus qu'avec du texte normal inséré avec `\mbox`.

```
\begin{equation}
  50 \textrm{ pommes} \times 100 \textbf{ pommes} =
  \textit{beaucoup de pommes}^2
\end{equation}
```

$$50\ \text{pommes} \times 100\ \mathbf{pommes} = \textit{beaucoup de pommes}^2 \qquad (1)$$

Cependant, comme cela est le cas pour LaTeX, il y a plus d'une façon de tondre un chat ! Il existe des commandes de modification de forme de police très semblables à celles utilisées précédemment, sauf qu'elles sont précisément conçues pour écrire du texte en mode mathématique. Alors pourquoi s'embêter à vous montrer comment utiliser `\textrm` et les autres si on peut trouver les équivalents en mode maths ? Bien, c'est parce qu'elles sont subtilement différentes. Les commandes de changement de forme de police en mode mathématique sont :

Commande	Forme de police	Exemple
`\mathrm{...}`	Roman	$\mathrm{e = mc^2}$
`\mathit{...}`	Italic	$\mathit{e = mc^2}$
`\mathbf{...}`	En gras	$\mathbf{e = mc^2}$
`\mathsf{...}`	Sans serif	$\mathsf{e = mc^2}$
`\mathtt{...}`	Typewriter	$\mathtt{e = mc^2}$
`\mathcal{...}`	Calligraphie	$\mathcal{] = \Updownarrow]^{\in}}$

Les commandes de changement de forme de police mathématique peuvent s'appliquer à une formule entière, et pas simplement un seul élément textuel : elles mettent en forme uniquement les lettres, les nombres, et les lettres majuscules grecques, mais le reste des objets mathématiques est ignoré. Ainsi, généralement, il vaut mieux employer les les commandes spécifiques de maths s'il y a lieu. Notez que l'exemple de calligraphie donne un rendu plutôt étrange. C'est parce que pour les lettres, il exige des caractères majuscules. Les lettres restantes sont transformées en symboles spéciaux.

Changement de la taille des formules

Probablement un événement rare, mais il pourrait arriver que vous préféreriez disposer d'une commande pour changer la taille des caractères. Par exemple, en utilisant le mode texte en mode mathématique, par défaut une simple fraction ressemblera à ceci $\frac{1}{2}$ et vous aimeriez certainement la voir plus grande, comme dans le mode mathématique isolé par exemple, comme ceci : $\displaystyle\frac{1}{2}$. Une solution simple est d'utiliser les tailles prédéfinies pour les objets mathématiques :

`\displaystyle`	Taille pour les formules en mode mathématique isolé
`\textstyle`	Taille pour les formules en mode texte
`\scriptstyle`	Taille pour les indices supérieurs/inférieurs
`\scriptscriptstyle`	Taille pour les indices supérieurs/inférieurs des indices

Un exemple classique pour illustrer cela est celui des fractions continues. Examinons le code suivant :

```
\begin{equation}
  x = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + a_4}}}
\end{equation}
```

Comme vous pouvez le voir, les fractions continues deviennent plus petites au fur et à mesure (bien qu'elles ne le deviennent pas indéfiniment dans l'exemple), et atteignent la taille limite de `\srciptscriptstyle`. Si vous voulez garder une taille acceptable, vous pouvez définir chaque fraction en utilisant le style mathématique isolé à la place.

```
\begin{equation}
  x = a_0 + \frac{1}{\displaystyle a_1
          + \frac{1}{\displaystyle a_2
          + \frac{1}{\displaystyle a_3 + a_4}}}
\end{equation}
```

Une autre solution est d'employer la commande `\DeclareMathSizes` permettant de sélectionner vos tailles préférées. Vous pouvez seulement définir des tailles pour les style `\displaystyle}`, `\textstyle`, etc. Un inconvénient de cette commande est que les tailles des caractères en mode mathématiques ne peuvent être fixées que globalement, car elle ne peut être utilisée que dans le préambule du document.

Cependant, il est assez facile d'employer : `\DeclareMathSizes{ds}{ts}{ss}{sss}`, où *ds* représente la taille d'affichage (*display size*), *ts* est la taille du texte (*text size*), etc. Les valeurs que vous donnez pour définir la taille sont supposées représenter un nombre de points (pt). Dans le [tutorial10/mathsize2.pdf example document (mathsize2.pdf)], les tailles des maths ont été rendues beaucoup plus grandes que nécessaire afin de mettre en évidence les changements intervenus. NOTe : les changements ont lieu uniquement si la valeur du premier paramètre correspond exactement à la taille courante des textes de document. Il est donc fréquent de voir un ensemble de déclarations communes dans le préambule,

lorsque la police principale est changée. Par exemple,

```
\DeclareMathSizes{10}{18}{12}{8}   % Pour un texte de taille 10
\DeclareMathSizes{11}{19}{13}{9}   %  Pour un texte de taille 11
\DeclareMathSizes{12}{20}{14}{10}  %  Pour un texte de taille 12
```

Équations multi-lignes (environnement `eqnarray`)

Imaginez que vous ayez une équation que vous vouliez la manipuler, par exemple, pour la simplifier. Souvent cela s'effectue après un certain nombre d'étapes en aidant le lecteur à comprendre comment obtenir à partir de l'équation originale le résultat final. Cela devrait être une tâche relativement simple, mais nous verrons que les commandes utilisées dans le cours précédent pour écrire des mathématiques ne sont pas adaptées. En utilisant ce que nous savons jusqu'ici nous avons : Le résultat obtenu est plutôt laid. Le moyen d'afficher les divers éléments en les alignant est d'utiliser une table, et d'y placer les équations en ligne. Allons y :

```
\begin{tabular}{ r l }
\(10xy^2+15x^2y-5xy\) & \(= 5\left(2xy^2+3x^2y-xy\right)\) \\
 & \(= 5x\left(2y^2+3xy-y\right)\) \\
 & \(= 5xy\left(2y+3x-1\right)\)
 \end{tabular}
```

Cela ne donne pas un résultat trop mauvais. Peut-être que l'espace supplémentaire entre le membre gauche de l'équation et le signe égal est plus grand que l'espace du côté droit et ne donne pas une apparence parfaite. Cela peut être rectifié en ajoutant une colonne supplémentaire et en mettant le signe égal dans la colonne centrale :

```
\begin{tabular}{ r c l }
  \(10xy^2+15x^2y-5xy\) & \(=\) & \(5\left(2xy^2+3x^2y-xy\right)\) \\
   & \(=\) & \(5x\left(2y^2+3xy-y\right)\) \\
   & \(=\) & \(5xy\left(2y+3x-1\right)\)
\end{tabular}
```

C'est beaucoup mieux. Un autre problème est que l'espace vertical entre les lignes fait apparaître les équations un peu trop serrées sur le côté droit. Il serait souhaitable d'ajouter un petit espace pour que les lignes dans la table soient un peu plus grandes. (vous devez ajouter `\usepackage{array}` dans votre préambule pour réaliser ce travail.)

% Augmente la taille verticalement. Nécessite le paquet tableau. Voir le préambule.

```
\setlength{\extrarowheight}{0.3cm}
```

Fichier:Multiligne4.png

Cependant, par ce moyen nous avons dû effectuer beaucoup trop de travail, et il reste un inconvénient important au bout du compte : vous ne pouvez pas numéroter les équations. Puisqu'une fois que vous êtes dans l'environnement `tabular`, vous pouvez seulement employer le type intégré d'affichage de maths. Pour la numérotation des équations, vous devez employer le paquet du mode d'affichage `equation`, mais vous ne pouvez pas dans cet exemple. C'est là que `eqnarray` devient extrêmement utile.

`eqnarray`, comme le nom le suggère provient du paquet `array}` qui est fondamentalement un environnement simplifié de `tabular`. Le paquet `array` fut introduit dans le [latextutorial9.html tutorial 9 (Mathematics I)] pour former des matrices. Voici un exemple d'utilisation d'`eqnarray`:

```
\begin{eqnarray}
  10xy^2+15x^2y-5xy & = & 5\left(2xy^2+3x^2y-xy\right) \\
   & = & 5x\left(2y^2+3xy-y\right) \\
   & = & 5xy\left(2y+3x-1\right)
\end{eqnarray}
```

Comme vous pouvez le voir, tout est bien présenté et organisé comme prévu. Bien, sauf que chaque rangée du tableau a reçu son propre numéro d'équation. Tandis que cette possibilité est utile dans certains cas, elle n'est pas requise ici, seulement un numéro suffira. Pour supprimer des numéros d'équation d'une rangée donnée, ajoutez la commande `\nonumber` juste avant la commande de fin de ligne (`\\`).

```
\begin{eqnarray}
  10xy^2+15x^2y-5xy & = & 5\left(2xy^2+3x^2y-xy\right) \nonumber \\
   & = & 5x\left(2y^2+3xy-y\right) \nonumber \\
   & = & 5xy\left(2y+3x-1\right)
\end{eqnarray}
```

Si vous ne voulez pas du tout de numéro d'équation, alors plutôt que d'ajouter la commande `\\nonumber` à chaque rangée, employez la version étoilée de l'environnement c'est-à-dire `\begin{eqnarray*}... \end{eqnarray*}`.

Note : il y a une limite de 3 colonnes dans l'environnement de `eqnarray`. Si vous avez besoin de davantage de flexibilité, vous ferez mieux d'utiliser les paquets de Maths AMS et l'environnement `align`. Pour des raisons typographiques, l'environnement `align` doit être préféré à l'environnement de `eqnarray`.

Casser les longues équations

D'abord voyons un exemple de longue équation (note : lorsque vous visualiserez l'exemple de document de cette rubrique, vous verrez que cette équation est en fait plus large que la largeur du texte. Ce n'est pas si évident que cela sur cette page Web)

Latex ne casse pas les longues équations pour les faire entrer dans les marges comme il le fait avec le texte normal. Ainsi, il vous appartient de composer l'équation convenablement (si elle déborde la marge). Ceci exige typiquement une certaine utilisation créatrice d'`eqnarray` afin d'obtenir des objets découpés sur une nouvelle ligne et bien alignés. Par exemple,

```
\begin{eqnarray*}
  \left(1+x\right)^n & = & 1 + nx + \frac{n\left(n-1\right)}{2!}x^2 \\
  & & + \frac{n\left(n-1\right)\left(n-2\right)}{3!}x^3 \\
  & & + \frac{n\left(n-1\right)\left(n-2\right)\left(n-3\right)}{4!}x^4 \\
  & & + \ldots
\end{eqnarray*}
```

Vous pouvez noter qu'à partir de la deuxième ligne, l'espace entre le plus initial et la fraction suivante est légèrement plus petit que la normale. C'est dû au fait que Latex traite les signes + et – de deux façons différentes. Le plus souvent il les considère comme un opérateur binaire. Quand deux objets mathématiques apparaissent de part et d'autre du signe, Latex suppose qu'il est un opérateur binaire, et attribue un certain espace de chaque côté du signe. La manière alternative est une désignation de signe; lorsque vous énoncez qu'une certaine quantité mathématique est positive ou négative. Comme en maths, on suppose que de tels objets sont positifs à moins qu'un signe le précède. Dans cet exemple, vous voulez que le signe apparaisse près de l'élément approprié pour montrer leur correspondance. C'est l'interprétation pour laquelle Latex a opté dans l'exemple ci-dessus. Pour ajouter la place nécessaire, vous pouvez insérer *un caractère invisible* en utilisant `{}`, comme nous le montrons ci-après :

```
\begin{eqnarray*}
  \left(1+x\right)^n & = & 1 + nx + \frac{n\left(n-1\right)}{2!}x^2 \\
  & & {} + \frac{n\left(n-1\right)\left(n-2\right)}{3!}x^3 \\
  & & {} + \frac{n\left(n-1\right)\left(n-2\right)\left(n-3\right)}{4!}x^4 \\
  & & {} + \ldots
\end{eqnarray*}
```

Une autre solution serait d'éviter ce problème en laissant le symbole + à l'extrémité de la ligne précédente plutôt que de le placer au début de la ligne courante :

Latex supporte une autre convention d'écriture des longues équations. C'est la manière qui est la plus fréquemment utilisée pour composer de longues équations dans les livres et les articles, et c'est évidemment ma manière préférée de les écrire. L'utilisation d'`eqnarray`, conjointement avec la commande `\lefteqn{... }` placée autour du contenu précédant le signe = donne le résultat suivant :

```
\begin{eqnarray*}
  \lefteqn{\left(1+x\right)^n = } \\
  & & 1 + nx + \frac{n\left(n-1\right)}{2!}x^2 + \\
  & & \frac{n\left(n-1\right)\left(n-2\right)}{3!}x^3 + \\
  & & \frac{n\left(n-1\right)\left(n-2\right)\left(n-3\right)}{4!}x^4 + \\
  & & \ldots
\end{eqnarray*}
```

Notez que la première ligne du `eqnarray` contient uniquement `\lefteqn`. Et avec cette commande, il n'y a aucun séparateur de colonne (`&`). La raison pour laquelle cette commande affiche les objets comme il se doit est que la commande `\lefteqn` imprime l'argument, en indiquant à Latex que la largeur est nulle. Le contenu de la première colonne étant vide, à l'exception de l'espace inter-colonne, cela impose le retrait des lignes suivantes.

Contrôle de l'espacement horizontal

Latex est évidemment un outil très puissant pour la composition de textes mathématiques. La composition de textes scientifiques était l'un des principaux objectifs du noyau du système Tex que Latex prolonge. Cependant, il n'interprète pas toujours exactement les formules comme vous aimeriez qu'il le fasse. Il doit parfois choisir entre plusieurs options quand il rencontre une expression ambiguë. Le résultat a tendance à être légèrement incorrect au niveau de l'espacement horizontal, mais reste malgré tout encore satisfaisant. Pourtant, tous les perfectionnistes voudront sans nul doute affiner leurs formules afin d'assurer un espacement horizontal impeccable. Ce sont généralement des ajustements très subtils.

Dans d'autres circonstances où Latex a fait son travail correctement, vous aimeriez juste ajouter un certain espace, pour éventuellement placer un commentaire. Par exemple, dans l'équation suivante, il est préférable de s'assurer qu'il y ait une place suffisante entre les maths et le texte.

```
\[f(n) = \left\{
\begin{array}{l l}
  n/2 & \quad \text{si $n$ est pair}\\
  -(n+1)/2 & \quad \text{si $n$ est impair}\\ \end{array} \right. \]
```

$$f(n) = \begin{cases} n/2 & \text{si } n \text{ est pair} \\ -(n+1)/2 & \text{si } n \text{ est impair} \end{cases}$$

Latex a défini deux commandes qui peuvent être employées n'importe où dans les documents (pas seulement en mode mathématique) pour insérer un certain espace horizontal. Ce sont les commandes `\quad` et `\qquad`.

Un `\quad` abréviation de quadratin est une espace égale à la taille de la police courante. Ainsi, si vous employez une police de 11 points (11pt), alors l'espace fourni par `\quad` sera également de 11 points (horizontalement, naturellement). Le `\qquad` donne deux fois l'espace précédent. Comme vous pouvez le voir à partir du code de l'exemple ci-dessus, des `\quad` ont été employés pour ajouter une certaine séparation entre les maths et le texte.

Pour revenir à l'affinement mentionné au début du document, un bon exemple montrerait une simple intégrale indéfinie de y par rapport à *x* :

$$\int y \mathrm{d}x$$

Si vous désirez essayer, vous pouvez écrire :

```
\[ \int y \mathrm{d}x \]
```

Cependant, ceci ne donne pas un résultat correct. Latex ne respecte pas l'espace blanc à gauche dans le code pour signifier que le *y* et le *x* sont des entités indépendantes. Au lieu de cela, il les colle l'une à l'autre. Un `\quad` laisserait un trop grand espace dans cette situation; il n'y a besoin que de quelques petits espaces dans ce type d'exemple, et c'est pour cela que Latex fournit :

Commande	Description	Taille
`\,`	petit espace	3/18 d'un quadratin
`\:`	moyen espace	4/18 d'un quadratin
`\;`	grand espace	5/18 d'un quadratin
`\!`	espace négatif	-3/18 d'un quadratin

NB vous pouvez utiliser plusieurs commandes à la suite pour laisser un plus grand espace si nécessaire.

Ainsi, pour corriger le problème actuel :

`\int y\, \mathrm{d}x` $\int y\, \mathrm{d}x$

`\int y\: \mathrm{d}x` $\int y\: \mathrm{d}x$

`\int y\; \mathrm{d}x` $\int y\; \mathrm{d}x$

Utiliser un espace négatif peut sembler anormal, cependant celui-ci n'existerait pas s'il était inutile. Prenez l'exemple suivant :

$$\left(\begin{array}{c} n \\ r \end{array} \right) = C_n^r = \frac{n!}{r!(n-r)!}$$

La notation matricielle pour représenter les coefficients binomiaux n'est pas assez « capitonnée ». Il y a trop d'espace entre les parenthèses et le contenu réel. Ceci peut facilement être corrigé en ajoutant quelques espaces négatifs après la parenthèse gauche et avant la parenthèse droite.

$$\left(\!\!\!\begin{array}{c} n \\ r \end{array}\!\!\!\right) = C_n^r = \frac{n!}{r!(n-r)!}$$

```
\[\left(\!\!\!
  \begin{array}{c}
    n \\
    r
  \end{array}
  \!\!\!\right) = C^r_n = \frac{n!}{r!(n-r)!}
\]
```

Introduction aux maths d'AMS

Cette section est encore en développement.

Quelques commandes LaTeX utiles sont disponibles sous forme de paquets nommés *amsmath*. Ils peuvent être chargés sur la plupart des installations de LaTeX en plaçant dans le préambule de votre document cette commande. (C'est-à-dire avant la commande `\begin{document}`) :

```
\usepackage{amsmath}
```

Commande align

Une commande très utile rend possible, au moyen de ces paquets, l'alignement des objets, par exemple le signe d'égalité (=) sur des lignes d'équations successives. Voici un exemple :

```
\begin{align}
  x &= a + (b + a) \\
    &= 2a + b.
\end{align}
```

Le symbole esperluette (`&`) est utilisé pour préciser les points dans chaque ligne qui seront supposés être alignés verticalement, mais n'apparaissent pas dans le résultat final :

$$\begin{aligned} x &= a + (b + a) && (1) \\ &= 2a + b. && (2) \end{aligned}$$

Les équations sont numérotées par défaut. Les numéros d'équation peut être supprimés en remplaçant le mot `align` par le mot `align*` au début et à la fin des équations à aligner :

```
\begin{align*}
  x &= a + (b + a) \\
    &= 2a + b.
\end{align*}
```

Le résultat ressemble à ceci :

$$\begin{aligned} x &= a + (b + a) \\ &= 2a + b. \end{aligned}$$

Une autre solution est de les supprimer sur une ligne particulière en ajoutant l'expression `\notag`, comme nous le montrons ici :

```
\begin{align}
  x &= a + (b + a) \notag \\
    &= 2a + b.
\end{align}
```

Le résultat ressemble à ceci :

$$\begin{aligned} x &= a + (b + a) \\ &= 2a + b. && (1) \end{aligned}$$

Des annotations dans chaque ligne peuvent être ajoutées en les séparant des équations, en utilisant deux esperluettes (`&&`) :

```
\begin{align}
  x &= a + (b + a) && \text{(Axiome C)} \\
    &= 2a + b && \text{(Axiomes A et F)}.
\end{align}
```

Voici le résultat :

$$\begin{aligned} x &= a + (b + a) && \text{(Axiome C)} \\ &= 2a + b && \text{(Axiomes A et F)}. \end{aligned}$$

Commande multline

Le paquet amsmath comporte également un environnement `multline` (et sa version non-numérotée `multline*`) qui permettent d'écrire une équation sur plusieurs lignes. Voici un exemple :

```
\begin{multline}
  \exp(x) = 1 + x + x^2 / 2 + x^3/ 3! + \cdots \\
            \cdots + x^n/ n! + \cdots
\end{multline}
```

Si on souhaite forcer manuellement la justification d'une partie de la formule, on peut utiliser les commandes `\shoveright` et `\shoveleft`. Exemple :

```
\begin{multline*}
  X = a + b + c + d + e + f + g + h + i + j + k + l + m \\
      \shoveright{+ n + o + p + q + r + s + t + u + v} \\
      +w+x+y+z+1+2+3+4+691
\end{multline*}
```

Résumé

Comme vous pouvez commencer à le voir, la composition de textes mathématiques peut parfois être embêtante. Cependant, parce que LaTeX fournit de nombreux contrôles, vous pouvez obtenir des documents mathématiques de qualité professionnelle avec relativement peu d'effort (une fois que vous vous êtes familiarisé avec LaTeX, bien sûr). Il serait possible d'aller plus loin sur le thème des mathématiques parce que cela semble illimité. Cependant, avec ces deux tutoriels, vous devriez être capable de vous en sortir honorablement.

Fichiers : [tutorial10/textineqn.tex textineqn.tex] | [tutorial10/textineqn.pdf textineqn.pdf] | [tutorial10/mathsize.tex mathsize.tex] | [tutorial10/mathsize.pdf mathsize.pdf] | [tutorial10/mathsize2.tex mathsize2.tex] | [tutorial10/mathsize2.pdf mathsize2.pdf] | [tutorial10/multiline.tex multiline.tex] | [tutorial10/multiline.pdf multiline.pdf] | [tutorial10/longeqns.tex longeqns.tex] | [tutorial10/longeqns.pdf longeqns.pdf] | [tutorial10/hspacing.tex hspacing.tex] | [tutorial10/hspacing.pdf hspacing.pdf]
ressources utiles : Guide d'utilisation du paquet `amsmath` [pdf (ftp://ftp.ams.org/pub/tex/doc/amsmath/amsldoc.pdf) [archive]]

Gestion des gros documents

La gestion des gros documents avec LaTeX s'articule sur trois niveaux :

- rigueur de la programmation ;
- choix d'une classe de document adaptée ;
- séparation du code source en plusieurs fichiers ;
- avoir un document final exploitable.

Rigueur de la programmation

Il faut appliquer à LaTeX la même rigueur que l'on utilise pour la programmation classique. En fait, tout a déjà été dit précédemment, mais nous rappelons ici ces éléments qui, s'ils sont toujours importants, deviennent capitaux dans le cas des gros documents :

- utiliser un éditeur de texte adapté, qui va faciliter l'écriture et éviter les erreurs ;
- structurer le code source, en n'hésitant pas à faire des retours de ligne et à mettre des commentaires :
  - limiter les lignes à 72 caractères,
  - pas plus d'une commande par ligne, mais on peut étaler une commande sur plusieurs lignes (penser à mettre un `%` en fin de ligne si les retours de lignes posent problème dans la commande),
  - mettre des commentaires détaillés pour repérer facilement les différentes sections, sous-sections, etc.
  - utiliser des décalages de colonnes (indentations par des tabulations) pour mettre en évidence l'imbrication des structures (listes, tableaux, …) ;
- séparer le fond de la forme ;
- regrouper le préambule dans un fichier séparé, et commenter les macros dans ce fichier.

Choix d'une classe de document adaptée

Il faut choisir une classe prévue pour les gros documents. Ces classes disposent en particulier du maximum de niveaux de sectionnement (7 niveaux de `\part` à `\subparagraph`).

On citera en particulier :

- `book` ;
- `report` ;
- `memoir`.

Les classes `book` et `memoir` disposent de commandes permettant de distinguer l'introduction, le corps et les annexes du document (`\frontmatter`, `\mainmatter` et `\backmatter`). Les annexes sont introduites par la commande `\appendix`.

Séparation du code source en plusieurs fichiers

Pour les gros documents, il faut scinder le code source en plusieurs fichiers. Cela permet de limiter les dégâts en cas de mauvaise manipulation, et de s'y retrouver plus facilement.

On crée donc plusieurs fichiers `.tex` :

- un fichier par chapitre, ou par section selon le niveau de sectionnement que l'on choisit ; dans ces fichiers annexes, il est inutile de rappeler la classe et les packages utilisés.
- un document « maître » faisant appels aux différents fichiers, dont la classe et les packages utilisés s'appliqueront à tous les fichiers utilisés.

Le document maître ne doit contenir que :

- l'appel de la classe : `\documentclass[`*`options`*`]{`*`classe`*`}` ;
- l'appel au préambule : `\input{`*`preambule`*`}` ;
- les commandes de page de titre : `\title{`*`titre`*`}`, `\author{`*`auteur`*`}`, `\date{`*`date`*`}`, `\maketitle` ;
- les commandes `\begin{document}` et `\end{document}` ;
- les appels aux fichiers sources ;
- éventuellement, les commandes de sectionnement `\front/main/backmatter` et `\appendix`, et les commandes de génération de tables et index : `\tableofcontents`, `\makeindex` et `\printindex`, `\bibliographystyle{`*`style`*`}` et `\bibliography{`*`fichierbib`*`}`, `\listoftables` et `\listoffigures`.

Les appels des fichiers source se font avec la commande `\include{`*`fichier`*`}{`*`fichier`*`}` ou `\input{`*`fichier`*`}` : `\include` provoque un saut de page et est plus adapté aux chapitres, `\input` est plus adapté aux sections, aux fichiers relativement petits.

Un fichier appelé ne peut pas lui-même contenir de `\include`, mais il peut contenir un `\input`.

Regrouper des articles

Dans le cas des publication scientifiques, chaque publication est un document (classe de document de type `article`) avec un titre, un résumé et le nom des auteurs, mais plusieurs articles sont regroupés dans un même ouvrage (périodique, compte-rendu de congrès, …). Pour faire ceci avec

LaTeX, on dispose de la classe `combine` [22] (http://tug.ctan.org/cgi-bin/ctanPackageInformation.py?id=combine)

Voir aussi http://www.tex.ac.uk/cgi-bin/texfaq2html?label=multidoc

Avoir un document final exploitable

Le document doit être agréable à lire, donc… lisible. On s'attachera en particulier à ajuster le nombre de colonnes de texte au corps afin d'avoir une soixantaine de caractères en largeur par colonne (entre 25 et 80).

Puisqu'il est volumineux, on doit pouvoir facilement retrouver une information. Il doit donc comporter :

- une table des matières et éventuellement un sommaire ;
- des références croisées ;
- un index ;
- une bibliographie.

Lorsqu'il y a beaucoup de références dans le document il peut être agréable de pouvoir les suivre dans le fichier `.pdf` à la manière de liens hypertextes. Dans ce cas on peut utiliser l'extension `\hyperref` qui construit les liens à la compilation (fonctionne avec `pdflatex` uniquement).

Faire des présentations

Nous allons présenter différentes classes LaTeX permettant de produire des présentations.

Beamer

Beamer est sans doute la solution la plus couramment utilisée pour faire des présentations sous LaTeX.

Premier exemple

La structure de base d'une présentation est :

```
\documentclass{beamer}

\begin{document}

\title{Titre de la présentation}
\maketitle

\begin{frame} % premier transparent
\frametitle{Titre du premier transparent}
\framesubtitle{Et son sous-titre}

Contenu du transparent.

\end{frame}

\begin{frame} % deuxième transparent
…
\end{frame}

\end{document}
```

Lors de la première compilation, le système de gestion des classes et extensions devrait télécharger automatiquement les fichiers permettant l'utilisation de `beamer`, ce qui inclut `pgf` et `xcolor`. Sinon, il faut aller les télécharger à la main à l'adresse

> https://bitbucket.org/rivanvx/beamer/wiki/Home

puis placer les fichiers obtenus dans les répertoires dédiés et lancer `texhash` pour mettre à jour la base de données LaTeX (voir *Installer des extensions supplémentaires*).

On peut y adjoindre de nombreux paramètres. Par exemple :

```
\documentclass{beamer}

\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{graphicx}
\usepackage[french]{babel}

\usetheme{Warsaw}

\begin{document}

\begin{frame}
\frametitle{Notion de groupe}
Un groupe est un ensemble $G$ muni d'une loi de composition interne $*$ telle que :
\begin{itemize}
\item La loi $*$ est associative;
\item La loi $*$ possède un élément neutre;
\item Tout élément de $G$ admet un symétrique pour la loi $*$.
\end{itemize}
\end{frame}

\end{document}
```

Fichier:Beamer premier exemple.jpg

Le préambule

On déclare la classe du document :

```
\documentclass[11pt, sans]{beamer}
```

L'option 'handout' permet de désactiver les animations (voir ci-dessous). C'est une option pratique pour l'impression.

```
\documentclass[11pt, sans, handout]{beamer}
```

Il y a trois sortes de thèmes :

- les thèmes généraux qui définissent l'ensemble des options,
- les thèmes intérieurs qui définissent le style du contenu du slide,
- les thèmes extérieurs qui définissent le style du cadre du slide (barre contextuelle, etc).

Les thèmes généraux

Un thème permet à Beamer de définir la position des menus, la couleur, les formes des blocs etc...

Liste des thèmes

- JuanLesPins
- Malmoe
- PaloAlto
- Berlin
- Boadilla
- Copenhagen
- Hannover
- Goettingen
- Montpellier
- Rochester
- Madrid
- Antibes
- Singapore
- Szeged
- Warsaw, thème par défaut
- Ilmenau
- Luebeck
- AnnArbor
- CambridgeUS
- Dresden

Les thèmes intérieurs

Pour avoir des objets anguleux :

```
\useinnertheme{rectangles}
```

Pour avoir des objets arrondis :

```
\useinnertheme{round}
```

Les thèmes extérieurs

Pour avoir une simple ligne indiquant les principales informations :

```
\useoutertheme{infolines}
```

Choisir l'ensemble des paramètres du thème

On peut définir soi-même les couleurs que l'on veut :

```
\definecolor{color1}{RGB}{33,33,33}
\definecolor{color2}{RGB}{222,69,0}
\definecolor{color3}{RGB}{239,239,239}
\definecolor{color4}{RGB}{0,119,170}
\setbeamercolor{structure}{bg=color1,fg=color2}
\setbeamercolor{normal text}{bg=color2,fg=color3}
\setbeamercolor{background canvas}{bg=color3}
\setbeamercolor{alerted text}{bg=color4}
```

- Astuce : Les codes couleurs peuvent être facilement obtenus en naviguant sur des pages internet grâce à l'extension colorzilla

(http://www.colorzilla.com/firefox/) [archive] de Firefox.

Pied de page

On peut ajouter le numéro du transparent courant et le nombre total de transparents avec la commande suivante :

```
\addtobeamertemplate{footline}{\insertframenumber/\inserttotalframenumber}
```

Titres/Auteurs

```
\title[Présentation]{Titre de la présentation}
\subtitle[\ldots]{Soutenance Mémoire}
\author[Dupont]{Toto Dupont}
\institute[Paris X]{Paris X Nanterre}
\date{\today}
```

Créer des diapositives

Les diapositives, ou transparents, ou *slides* en anglais, sont définies par l'environnement `frame` : une diapositive commence par `\begin{frame}` et finit par `\end{frame}`. Il est aussi possible d'encapsuler toute la diapositive dans une commande `\frame{…}` (c'était d'ailleurs la première syntaxe).

Le titre de la diapositive est donné par `\frametitle{`*`titre`*`}`. On peut aussi indiquer un sous-titre avec `\framesubtitle{`*`sous-titre`*`}`

La structure globale d'un transparent est donc :

```
\begin{frame}
   \frametitle{…}

\end{frame}
```

ou bien

```
\frame{
   \frametitle{…}

}
```

Créer le plan

Pour afficher le plan on utilise la commande `\tableofcontents`.

- L'option `currentsubsection` permet de mettre en valeur la sous-section courante.
- L'option `currentsection` permet de mettre en valeur la section courante.
- '`sectionstyle=show/shaded`' détermine le mode d'apparition des sections et des sous-sections mises en valeur.

```
\frame{
   \frametitle{…}
   \tableofcontents[currentsubsection,sectionstyle=show/shaded,subsectionstyle=show/shaded/hide]
}
```

Créer des animations

Commande \pause

Commande \uncover

Liens externes

- Site officiel (https://bitbucket.org/rivanvx/beamer/wiki/Home/) [archive]
- Documentation officielle (en anglais) (http://mirrors.ctan.org/macros/latex/contrib/beamer/doc/beameruserguide.pdf) [archive]

LaTeX pour enseignants

LaTeX a bien entendu des fonctions spécifiques pour les enseignants. Par exemple, la classe **exam** permet de concevoir des examens et des feuilles d'exercices. Il existe aussi d'autres classes comme **probsoln** ou **mathexm** ou des packages comme **exercise**. Pour les QCM avec auto correction en latex voir le logiciel auto-multiple choice http://home.gna.org/auto-qcm/.

La classe exam

La classe **exam** est très pratique pour concevoir des examens ou des feuilles d'exercices. Les solutions sont incluses dans un environnement spécifique. Elles peuvent être imprimées ou non selon les options choisies dans le préambule. La classe **exam** permet aussi de compter le total de points inclus dans l'examen.

Le préambule

Dans le préambule, on peut préciser les lignes suivantes :

```
\documentclass[a4paper,11pt]{exam}
\printanswers % pour imprimer les réponses (corrigé)
% \noprintanswers % Pour ne pas imprimer les réponses (énoncé)
\addpoints % Pour compter les points
% \noaddpoints % pour ne pas compter les points
\qformat{\textbf{Question\thequestion}\quad(\thepoints)\hfill} % Pour définir le style des questions (facultatif)
\shadedsolutions % définit le style des réponses
% \framedsolutions % définit le style des réponses
\usepackage{color} % définit une nouvelle couleur
\definecolor{SolutionColor}{rgb}{0.8,0.9,1} % bleu ciel
\renewcommand{\solutiontitle}{\noindent\textbf{Solution:}\par\noindent} % Définit le titre des solutions
```

On peut remplacer les trois premières lignes par la ligne suivante.

```
\documentclass[a4paper,11pt,answers,addpoints]{exam}
```

Document

L'examen est inclus dans l'environnement **questions**. La commande **\question** introduit une nouvelle question. Le nombre de points est inclus entre crochets. Les solutions sont comprises dans l'environnement **solution**.

```
\begin{questions} % Début de l'examen
\question[2] Quelle est la solution ? % Nouvelle question à 2 points
\begin{solution}
C'est la solution
\end{solution}
\question[5] Qu'en pensez-vous ?
\begin{solution}
C'est mon opinion
\end{solution}
\end{questions}
```

Il est possible de personnaliser le titre de la question avec **\titledquestion{Titre de la question}[]**.

Introduction

Dans l'introduction on peut utiliser la macro **\numquestions** qui donne le total du nombre de questions et **\numpoints** qui donne le total du nombre de points.

```
\begin{minipage}{.8\textwidth}
Cet examen comprend \numquestions\ questions sur un total de \numpoints\ points.
\end{minipage}
```

Le backslash après **\numquestion** permet de forcer l'impression d'un espace entre la macro et le mot suivant.

Arts et loisirs

1. Écrire de la musique
2. Dessiner des échiquiers

Écrire de la musique

Il existe un langage libre et gratuit pour écrire de la musique : LilyPond. Il est assez semblable à LaTeX dans sa forme, et très performant. Avec, vous pouvez créer une partition, avec titre, texte de paroles, commentaires, … et générer :

- un fichier PS et PDF de la partition ;
- un fichier son Midi du morceau ;
- un fichier image, qui peut par la suite être intégré dans un fichier LaTeX.

Nous vous invitons vivement à utiliser LilyPond, d'autant plus que le code LilyPond peut être intégré directement dans le code LaTeX.

Toutefois, avant le développement de LilyPond, une extension fut développée pour permettre d'écrire de la musique avec LaTeX : MusicTeX, remplacée depuis par MusiXTeX. Il peut être plus pratique, pour de courts extraits, d'utiliser MusiXTeX plutôt que d'intégrer une image. Nous allons donc voir quelques fonctionnalités.

Notons toutefois que LilyPond est « orienté musique » : l'auteur a juste à penser au morceau, et écrit les voix séparément (on a une « écriture horizontale »), le programme se charge d'aligner les mesures, et les notes dans les mesures. On a une bonne séparation du fond et de la forme. À l'inverse, MusiXTeX est « orienté typographie » : l'auteur doit penser à la mise en forme, il doit prévoir l'alignement en regroupant ensemble les éléments des différentes voix (on a une « écriture verticale »). Mais comme le font remarquer les auteurs de MusiXTeX, « si cela vous paraît compliqué, rappelez vous que TeX a été conçu pour écrire des textes et non de la musique… » *(If this sounds complicated, remember that TeX was designed to typeset text and not music…)*. Toutefois, le résultat obtenu avec MusiXTeX est irréprochable.

Mode mathématiques et fonte Text Companion

Le mode mathématiques (introduit, en ligne, par `$ … $`), permet d'afficher quelques symboles musicaux :

- `\flat` : bémol ♭ ;
- `\natural` : bécarre ♮ ;
- `\sharp` : dièse ♯.

On remarque au passage que le dièse « ♯ » est différent du croisillon « # », appelé *hash sign* en anglais (et abréviation de « numéro » dans cette langue).

La fonte Text Companion est incluse dans le codage TS1. On en dispose donc en mettant dans le préambule

```
\usepackage{textcomp}
```

On peut alors écrire une croche avec l'instruction `\textmusicalnote`. On peut obtenir le C barré avec `\textcolonmonetary` (bien qu'il s'agisse ici d'un symbole monétaire).

LilyPond

Il existe un wikilivre sur LilyPond. Nous allons ici rappeler quelques généralités.

LilyPond est un projet totalement indépendant de LaTeX. Il doit être téléchargé à part, et les fichiers Lilypond possèdent leur propre compilateur.

Les fichiers LilyPond sont des fichiers texte portant l'extension `.ly`. Ils sont compilés par le programme `lilypond`. Le programme est livré avec un éditeur de texte, `lilypad`.

Éléments de base

L'élément de base de Lilypond est la phrase musicale, sous la forme :

```
{ notes }
```

Les *notes* sont de la forme *hauteur durée*, où

- la *hauteur* est indiquée
  - avec la notation anglosaxonne (`c d e f g a b` pour *do* à *si*),
  - la note est diésée en ajoutant `is` et bémolée en ajoutant `es` (`cis` pour un do dièse),
  - on monte d'une octave avec une apostrophe `'`, et on descend avec une virgule `,` (par exemple « `c'` » pour avoir le do une octave au dessus, `c,` pour une octave en dessous),
  - un silence est indiqué par un « `r` » *(rest)* ;
- la *durée* est indiquée
  - par un nombre selon la notation habituelle : `1` pour une ronde, `2` pour une blanche, `4` pour une noire, …
  - on met un point `.` derrière pour pointer la durée,
  - si l'on n'indique pas la durée, on utilise la même durée que la note précédente.

Par exemple

```
c4. r4.
```

pour un *do* noire pointée suivi d'un soupir pointé,

```
\relative c'{c8 d e c d4 d8 e f4 f e e}
```

pour le début de *J'ai du bon tabac*.

Un fichier peut contenir uniquement une ou plusieurs phrases musicales. Toutefois, un fichier contient en général d'autres informations, notamment concernant la mise en forme.

Accords et polyphonie

Les notes d'un accord se notent entre deux crochets pointus <…>. La durée se met après. Par exemple :

```
<c e g>4
```

Lilypond reconnaît également la notation anglo-saxonne des accords, comme par exemple `c:m` pour *do* mineur. La durée se met avant le deux-points : `c4:m`.

Si l'on veut faire figurer deux voix sur la même portée, on met les deux voix entre des doubles crochets <<…>>, et on les sépare par deux contre-obliques :

```
<<
   {voix une} \\
   {voix deux}
>>
```

Si l'on veut avoir plusieurs portées, on utilise :

```
<<

   \new Staff {voix une}

   \new Staff {voix deux}
>>
```

Format général

Un fichier `.ly` est un fichier texte de la forme :

```
\version version

\header {
   title = titre de l'oeuvre
   composer = nom du compositeur
}

\score {
   partition
}
```

où

- *version* est le numéro de version de lilypond (permet de faciliter les mises à jour) ;
- *titre de l'oeuvre* et *nom du compositeur* seront affiché en haut de la page ;
- *partition* est la description de la musique elle-même.

Si vous voulez juste un petit bout de partition, le plus simple est peut-être de retoucher l'image avec un logiciel, comme par exemple The Gimp, pour extraire la partie qui vous intéresse.

La partition est de la forme :

```
\new Staff \relative c' {
   \key c \major
   \clef G
   \time 4/4
```

```
    \tempo 4=60

    notes

}
```

où

- `\new Staff` indique une nouvelle portée ;
- `\relative` indique que l'octave est déterminée par la hauteur de la note précédente ; le `c'` indique que l'octave de référence est fixée par le do medium ;
- `\key c \major` indique que l'on est en *do* majeur ;
- `\clef G` indique que l'on place une clef de sol ;
- `\time 4/4` indique une mesure à $\frac{4}{4}$ (ou **C**) ;
- `\tempo 4 = 60` indique que le tempo est de 60 à la noire (`4`).

Notons que la déclaration de la tonalité (avec `\key`) place l'armure d'altérations. Si vous êtes en *sol* majeur (`\key g \major`) et que vous mettez `fis`, l'altération ne s'affichera pas puisqu'elle est à la clef. Si par contre vous mettez `f`, cela affichera un *fa* bécarre.

Compilation

L'éditeur `lilypad` permet de lancer la compilation :

- avec le menu **Compile | Typeset file**,
  — ou bien —
- avec la combinaison de touches Ctrl+R sous Windows, +R sous MacOS X.

On peut également, à partir du gestionnaire de fichiers (*Explorateur* Windows, *Finder* de MacOS X, *X-Window* d'Unix), faire un glisser-lâcher de l'icône du fichier `.ly` sur l'icône du programme `lilypond`. On peut enfin utiliser la ligne de commande

```
lilypond nom_du_fichier
```

Par défaut, Lilypond génère un fichier PostScript (`.ps`). Les PDF peuvent être intégrés directement dans fichier LaTeX.

Pour que Lilypond génère un fichier d'un format différent, compilez avec les options :

- `lilypond --pdf nom_de_fichier` pour avoir un PDF ;
- `lilypond -f=ps nom_de_fichier` pour avoir un PostScriptS ;
- `lilypond -f=png nom_de_fichier` pour avoir un PNG.

On peut aussi utiliser `--format=` au lieu de `-f=`.

Intégration du code LilyPond dans le code LaTeX

On peut intégrer directement du code LilyPond dans le source LaTeX ; le fichier devra être compilé avec `lilypond-book` avant d'être compilé par `latex`.

Le code LilyPond est écrit :

- soit dans un environnement `lilypond` ;
- soit dans le bloc d'une commande `\lilypond{…}` ;
- soit dans un fichier extérieur appelé dans le document LaTeX par `\lilypondfile{nom_de_fichier}`.

Sous MacOS X, le programme `lilypond-book` est dans le répertoire `…/LilyPond.app/Contents/Resources/bin`, voir *Introduction à LilyPond, Principe général > Programme en ligne de commande*.

Voir

- *Lilypond-book: Integrating text and music (http://lilypond.org/doc/v2.10/Documentation/user/lilypond /LilyPond_002dbook#LilyPond_002dbook)* *[archive]* ;
- pour l'utilisation de `lilypond-book` sous Microsoft Windows 95, 98 et ME : *LilyPond sous Cygwin (http://lilypond.org/web/install /windows.fr.html)* *[archive]*

MusiXTeX

Pour utiliser MusiXTeX, il faut mettre en préambule

```
\usepackage{musixtex}
```

Un morceau de musique a la forme générale :

```
\begin{music}
   \startextract
      \notes notes \enotes
   \endextract
\end{music}
```

où

- `\startextract` et `\endextract` indiquent le début et la fin de la portée ; pour les longs morceaux, on utilise `\startpiece` et `\endpiece` ;
- `\notes` et `\enotes` indiquent le début et la fin d'une phrase musicale.

Une partition peut contenir plusieurs phrases musicales ; elle sera donc une succession de `\notes … \enotes`. On peut utiliser `\en` à la place de `\enotes`.

Les notes sont indiquées sous la forme `\durée hauteur`, ou, si plusieurs notes ont la même durée, sous la forme `\durée{ hauteur_1 hauteur_2 … hauteur_n }`.

Par exemple,

```
\notes \qa c \enotes
```

ou

```
\notes \qa{c} \enotes
```

désignent un *do* (`c`) noire (`\qa`).

Remarque : muxsiXTeX est du LaTeX, il interprète donc les espace et les lignes vides. Les espaces et lignes vides peuvent donc avoir un effet important sur le rendu final ; on ne peut pas mettre en forme le code comme on veut. En particulier, **les lignes vides sont à proscrire**, mais on peut utiliser une ligne de commentaire. Pour la même raison, il peut être nécessaire de commenter les fins de lignes, notamment dans la zone d'en-tête de la partition.

Lors de la compilation, `latex` crée un fichier `.mx1`.

Taille et espacement

La dimension de la musique (taille des notes et des portées) est indiquée par une instruction, placée hors du contexte musical (par exemple en début de document). Par ordre croissant, on a :

- `\smallmusicsize` ;
- `\normalmusicsize` ;
- `\largemusicsize` ;
- `\Largemusicsize`.

Nous avons ci-dessus parlé de l'instruction `\notes`, mais il existe plusieurs instructions d'introduction permettant de faire varier l'espacement, par ordre croissant d'espacement :

- `\znotes` *(zero)* : pas d'espacement ;
- `\notes` : recommandé pour l'écriture des double-croches ;
- `\notesp` *(pointed)* : recommandé pour l'écriture des double-croches pointées ;
- `\Notes` : ~ croches ;
- `\notesp` ;
- `\NOtes` : ~ noires ;
- `\NOtesp` ;
- `\NOTes` : ~ blanches ;
- `\NOTesp` ;
- `\NOTEs` : ~ rondes.

On peut rajouter une espace après une note avec `\sk`.

Durées

Les durées sont :

- `\wh` : ronde *(whole note)* ;
- `\ha` : blanche, le placement de la queue (haut ou bas) étant laissé à l'appréciation de MusiXTeX *(half note automatic)* ;
- `\qa` : noire *(quarter note automatic)* ;
- `\ca` : croche *(*croche[1] *automatic)* ;

- `\cca` : double croche ;
- `\ccca` : triple croche.

On peut forcer l'orientation de la queue en remplaçant le *a* par

- *u* pour une hampe vers le haut *(up)* ;
- *l* pour une hampe vers le bas *(low)* ;

par exemple `\qu`, `\cl`.

Pour pointer une note, on ajoute `p`, et `pp` pour une double pointée. Par exemple, `\qap` indique une noire pointée.

Les silences sont :

- `\pause` : pause ;
- `\hpause` : demi-pause ;
- `\soupir` ou `\qp` : soupir ;
- `\ds` : demi-soupir ;
- `\qs` : quart de soupir ;
- `\hs` : huitième de soupir ;
- `\qqs` : seizième de soupir.

Les silences peuvent aussi être pointés, par exemple `\soupirp` pour un soupir pointé.

Hauteur

La notation est basée sur la notation anglosaxonne pour les notes lagrave à medium, qui sont notées `abcdefg`. Puis, les notes suivent l'ordre alphabétique : `h` pour *la*, `i` pour *si*, `j` pour le *do* aigu, …

Les notes des octaves inférieures sont notées en capitales : `CDEFG` pour les notes *do* à *sol* les plus graves : le *do* `C` est situé à deux lignes supplémentaires sous la portée en clef de *fa* 4e ligne. On a donc la succession des octaves

```
do  ré  mi  fa  sol la  si
C   D   E   F   G   H   I
J   K   L   M   N   a   b
c   d   e   f   g   h   i octave medium
j   k   l   m   n   o   p
q   r   s   t   u   v   w
x   y   z
```

On a aussi la possibilité de monter d'une octave avec une apostrophe `'`, ou de descendre une octave avec un accent grave `` ` ``. Cette montée ou cette descente sont définitifs. Par exemple, les deux déclarations suivantes sont identiques :

```
\notes \qa{CDEFGHI JKLMNab cdefghi jklmnop} \enotes
```

et

```
\notes \qa{``cdefghi 'cdefghi 'cdefghi 'cdefghi} \enotes
```

en effet, le `` `` `` initial fait descendre de deux octaves, puis on monte d'une octave à chaque fois.

Liens entre les notes

Utilisées telles quelles, les « notes à crochet » (croches, double-croches, triples-croches, quadruple-croches) sont écrites séparément. Quand elles se succèdent, on les lie en général par une « barre » située au-dessus ou en-dessous.

Si l'on veut introduire un lien placé au-dessus entre les notes de type croches, on utilise

- `\ibu`*`ref note pente`* (ibu pour *initiate beam up*), où
  - *ref* est la référence du lien : c'est un numéro arbitraire qui permet de démêler les différents liens qui s'étalent sur plusieurs `\notes … \enotes`,
  - *note* est la hauteur de la première note du lien, le lien est placé trois lignes au dessus de cette note,
  - *pente* est la pente du lien, entre `-9` (lien descendant) et `9` (lien montant), `0` indiquant un lien horizontal ;
- les notes ont pour durée `\qb` *`ref`* *(quarter beam ?)*, qui place la tête noire et la queue sans placer de crochet ;
- `\tbu`*`ref`* *(terminate beam)*, qui termine le lien, et se place *avant* la dernière note.

Pour les notes pointées, on utilise `\qbp`, et `\qbpp` pour les doubles-pointées. Pour les notes sans espace (par exemple pour les accords), on utilise le durée `\zqb`.

Par exemple

```
\Notes \ibu0d0\qb0{c c c}\tbu0\qb0d \enotes
```

Les autres liens sont introduits par :

- `ibl` *(initiate beam low)* : croches, lien en dessous ;
- `ibbu` et `ibbl` : double-croches ;
- `ibbbu` et `ibbbl` : triple-croches ;
- `ibbbbu` et `ibbbbl` : quadrule-croches.

On peut faire varier la multiplicité du lien en cours de lien, par exemple pour faire « croche — deux double-croches » :

```
\Notes \ibu0d0 \qb0{c} \nbbu0 \qb0{c} \qb0c \enotes
```

Les instructions sont :

- pour augmenter la multiplicité du lien :
  - `\nbbu`*`ref`* et `\nbbl`*`ref`* : lien double (double-croches),
  - `\nbbbu`*`ref`* et `\nbbbl`*`ref`* : lien triple (triple-croches),
  - `\nbbbbu`*`ref`* et `\nbbbbl`*`ref`* : lien quadruple (quadruple-croches) ;
- pour diminuer la multiplicité :
  - `\tbbbbu`*`ref`* et `\tbbbbl`*`ref`* : termine un lien quadruple, qui devient triple ;
  - `\tbbbu`*`ref`* et `\tbbbl`*`ref`* : termine un lien triple, qui devient double ;
  - `\tbbu`*`ref`* et `\tbbl`*`ref`* : termine un lien double, qui devient simple.

Si l'on fait varier la multiplicité et que l'on termine le lien en même temps, cela crée un lien multiple partiel. Par exemple, pour faire « croche pointée — double-croche » :

```
\Notes \ibu0e0\qbp0e\tbbu0\tbu0\qb0e \enotes
```

pour faire « double-croche — croche pointée » :

```
\Notes \ibbu0e0\roff{\tbbu0}\qb0e\tbu0\qbp0e \enotes
```

Le `\roff` sert à décaler la fonction vers la droite *(right offset)*, ce qui permet de donner une longueur non nulle au double-lien initial.

triolets, sextuolets et *n*-uolets

Il faut écrire toutes notes du *n*-uolet dans la même phrase (`\notes`…`\en`), et cette phrase ne doit contenir que ces notes-là.

Pour un triolet, on introduit la phrase par `\triolet` (avant la première note) ; on indique en argument la hauteur de note à laquelle placer le nombre. Par exemple

```
\triolet n
```

indique que le « 3 » sera placé à la note *n*, soit au niveau du *sol* aigu.

Pour un sextuolet, on utilise `\xtuplet6` suivi de la hauteur de note. On peut en fait mettre n'importe quel nombre après `\xtuplet`, pour avoir les différents *n*-uolets.

Accords

Un accord, c'est en fait déclarer des notes non suivies d'un espace. On utilise donc les durées :

- `\zw` pour une ronde *(zero-spacing whole note)* ;
- `\zhu` ou `\zhl` pour une blanche ;
- `\zqu` ou `\zql` pour une noire ;
- `\zcu` ou `\zcl` pour une croche ;
- `\zccu`, `\zcccu` ou `\zccl`, `\zcccl` pour une double ou triple croche.

Par exemple, pour l'accord parfait de *do* majeur en noires :

```
\zqu c \zqu e\qu g
```

ou

```
\zqu{ce}\qu g
```

ou

```
\zqu{ceg}\sk
```

La dernière note doit être une note avec espace, sans quoi la note suivante ferait aussi partie de l'accord, ou alors il faut faire suivre l'accord d'un espace. D'autre part, il ne faut pas d'espace avant la déclaration de durée de la dernière note de l'accord : cet espace provoquerait un décalage de la dernière note vers la droite.

Si l'accord contient des secondes, il est nécessaire de placer une notes de l'autre côté de la hampe. On utilise pour cela :

- `\rw` *(right whole note)* : ronde poussée vers la droite ;
- `\lw` *(left whole note)* : ronde poussée vers la gauche ;
- `\rh` et `\lh` : blanche ;
- `\rq` et `\lq` : noire ;

Phrasé et liaisons

Le principe du phrasé *(legato)* et de la liaison, parenthèse horizontale, est similaire à celui des liens. On débute un phrasé ou une liaison par

- `\islurd` *`ref note`* *(initiate slur down)* pour une liaison située en dessous ;
- `\isluru` *`ref note`* *(initiate slur up)* pour une liaison située au-dessus ;

où

- *ref* est un nombre permettant de suivre la parenthèse si l'on change de groupe `\notes … \enotes` ;
- *note* est la hauteur à laquelle commence le phrasé (en général la première note).

Le phrasé se termine par

```
\tslur ref note
```

où *note* est la hauteur de fin du phrasé. Par exemple

```
\Notes \islurd0c\qu{c d e}\tslur0f\qu f \enotes
```

Mesures

Le chiffrage de la mesure s'indique avec la fonction

```
\generalmeter{chiffrage}
```

où *chiffrage* est :

- `\meterfrac{`*a*`}{`*b*`}` pour indiquer la mesure *a*/*b* ;
  par exemple `\generalmeter{\meterfrac{3}{4}}`
- `\meterC` pour un **C** (**4/4**) ;
- `\allabreve` pour un C barré.

Les barres de mesure sont introduites par la fonction `\bar`. On peut placer des barres de répétition avec `\leftrepeat` (début du passage à répéter), `\rightrepeat` (fin du passage) et `\leftrightrepeat` (fin d'un passage et début du suivant).

Si l'on veut mettre des notes avant l'indication du chiffrage, on utilise `\meterskip` ; il prend fin à la première indication `\bar`.

Armure

Par défaut, la portée comporte une clef de *sol*. On peut indiquer une autre clef avec la commande

```
\setclef{1}{code}
```

le « 1 » indique qu'il s'agit de la première (et pour l'instant seule) voix. Le *code* est un chiffre :

- `0` pour une clef de *sol* ;
- `1` à `4` pour une clef d'*ut* 1 à 4 ;
- `5` pour une clef de *fa* 3 ;
- `6` pour une clef de *fa* 4.

Les altérations à l'armure s'indiquent par

```
\generalsignature{n}
```

où *n* est un entier :

- positif : c'est le nombre de dièses ;
- négatif : c'est le nombre de bémols.

Par exemple, `\generalsignature{1}` pour un morceau en *sol* majeur ou *mi* mineur (un dièse à la clef), `\generalsignature{-1}` pour un morceau en *fa* majeur ou *ré* mineur (un bémol à la clef).

Indications d'interprétation

Les indications d'interprétation, comme les accents, les ornements et les nuances, sont en général traités comme des durées non suivies d'espace. Par exemple, pour placer un *sforzando* au dessus de l'emplacement d'un do aigu, on écrit `\usf j`, suivi de la note sur lequel il est placé.

Les commandes sont :

- accents : les commandes commencent par *u (up)* si on les met au-dessus de la tête de note et *l (low)* si on les met en-dessous ; elles commencent par *bu (beam up)* si on les met au-dessus du lien entre les notes, et par *bl (beam low)* si on les met en-dessous :
  - `\upz`, `\lpz`, `\bupz` et `\blpz` : pizzicato (·) ;
  - `\usf`, `\lsf`, `\busf` et `\blsf` : sforzando (>) ;
  - `\ust`, `\lst`, `\bust` et `\blst` : staccato (—) ;
  - `\usfz` et `\busfz` : accent (∧) ;
  - `\lsfz` et `\blsfz` : accent (∨) ;
- ornements
  - `\mordent`, `\Mordent`, `\shake`, `\Shake` : mordant ;
  - `\turn`, `\backturn` : gruppetto ;
  - `\fermataup`, `\fermatadown`, `\Fermataup`, `\Fermatadown` : point d'orgue.

Pour indiquer une trille, on place

```
\Trille hl
```

où *h* est la hauteur de note où placer l'indication, et *l* est la longueur en nombre de notes. On peut aussi commencer la trille par `\ITrille nh`, où *n* est le numéro de la trille entre 0 et 6 (on peut placer plusieurs trilles si l'on a plusieurs voix), et *h* est la hauteur de note où placer l'indication ; on termine alors la trille par `\TTrille`.

Pour mettre des appogiatures, on utilise de petites notes, introduites par `\smallnotesize` ou `\tinynotesize`, et conclues par l'indication `normalnotesize`. Dans le cas d'une note unique sous la forme d'une croche barrée, on utilise `\grcu` et `\grcl`.

Les nuances sont `\ppp` *(il piu pianissomo possibile)*, `\pp` *(pianissimo)*, `\p` *(piano)*, `\mp` *(mezzo piano)*, `\mf` *(mezzo forte)*, `\f` *(forte)*, `\ff` *(fortissimo)* et `\fff` *(il piu fortissimo possibile)*. Pour un crescendo ou un decrescendo, on utilise `\crescendo{l}` et `\decrescendo{l}`, ou *l* est la longueur (par exemple *n*`\noteskip` pour *n* notes). Ces signes sont placés par la commande `\zcharnote h{texte}` où *h* est la hauteur de note et *texte* est l'indication.

Polyphonie

La numérotation des portées et des instruments se fait de bas en haut. La portée n° 1 est celle du bas.

Pour placer plusieurs portées, on débute l'environnnement par l'instruction

```
\instrumentnumber{n}
```

où *n* est le nombre d'instruments. Si un instrument utilise plusieurs portées (par exemple le piano), on précise

```
\setstaffs i{m}
```

où *i* est le numéro de l'instrument, et *m* est le nombre de portées. Puis, on indique les noms des instruments, avec l'instruction

```
\setname i{nom}
```

où *i* est le numéro de l'instrument. Si les noms sont longs, ils peuvent empiéter sur le début de la portée ou l'accolade qui réunit plusieurs portées. Pour éviter ceci, on peut augmenter l'alinéa avec

```
\parindent longueur
```

par exemple `\parindent 10mm`.

Puis, on découpe le morceau phrase par phrase — une phrase étant en général une portion de mesure choisi en fonction de l'alignement —, en séparant les instruments par un code>& :

```
\notes instrument 1 & instrument 2 & … \enotes
```

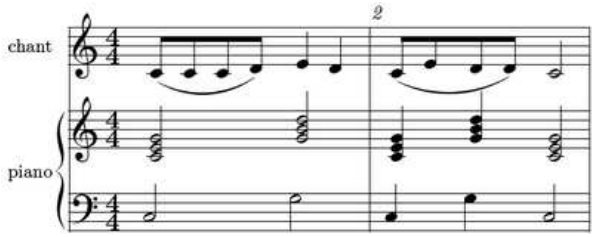
s'il y a un instrument avec deux portées, on les sépare par un tube `|`, par exemple :

```
\notes instrument 1 portée 1 | instrument 1 portée 2 & instrument 2 & … \enotes
```

Exemple

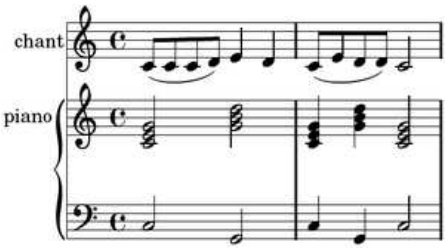
```
\documentclass[10pt]{article}

\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{musixtex}
\usepackage[frenchb]{babel}

\begin{document}

\normalmusicsize

\begin{music}

\instrumentnumber{2} % 2 instruments

\setstaffs 1{2} % instrument 1 (en bas) : 2 portées

\setclef{1}{60} % clef de fa (6) en 1, clef de sol (0) en 2
\generalmeter{\meterfrac{4}{4}} % mesure 4/4
\setname 1{piano} %
\setname 2{chant} %
\parindent 10mm % pour éviter la collision de piano avec l'accolade
\startextract
% {{bleu|1re mesure}}
\Notes
\ha J | % chgt portée, même instr
\zhu{c e}\hu g & % chgt instr ; pas d'espace entre } et \hu
\islurd0c\ibu0d0\qb0{c c c}\tslur0d\tbu0\qb0d
\enotes % assure l'alignement
%
\Notes
\ha N | % chgt portée, même instr
\zhu{g i}\hu k & % chgt instr
\qa{e d}
\enotes
\bar % après toutes les notes de la mesure, pour toutes les voix
% {{bleu|2e mesure}}
\Notes
\qa J |
\zqu{c e}\qu g &
\islurd0c\ibu0d0\qb0{c e}
\enotes
%
\Notes
\qa N |
\zqu{g i}\qu k &
\qb0{d}\tslur0d\tbu0\qb0d
\enotes
%
\Notes
\ha J |
\zhu{c e}\hu g &
\ha c
\enotes
%
\endextract

\end{music}

\end{document}
```

donne :



Par comparaison, voici le code Lilypond et le résultat :

```
\version "2.8.7"

\paper {
   #(define dump-extents #t)
   indent = 10\mm
   ragged-right = ##t
}

<<
   \new Staff \relative c' {
      \set Staff.instrument="chant"
      \clef G
      \time 4/4
      c8[( c c d]) e4 d4
      c8[( e d d]) c2
   }
   \new PianoStaff <<
      \new Staff \relative c' {
         \set Staff.instrument="piano"
         \clef G
         \time 4/4
         <c e g>2 <g' b d>
         <c, e g>4 <g' b d> <c, e g>2
      }
      \new Staff \relative c {
         \clef F
         \time 4/4
         c2 g
         c4 g c2
         }
   >>
>>
```



Paroles

MusiXTeX ne gère pas très bien l'alignement des paroles avec les notes. Il est recommandé de faire appel pour cela à l'extension `musiclyr`. Toutefois, MusiXTeX dispose tout de même des fonctions héritées de MusicTeX, avec lequel l'alignement était correct. Le problème est que les caractères — leur chasse et leur approche — ne sont pas étirables ou contractables alors que la largeur et l'espacement des notes peut varier.

La première chose consiste à réserver de la place verticale. Pour cela, on utilise

```
\setinterinstrument n{longueur}
```

qui crée un « interinstrument » *au dessus* de l'instrument *n*. La *longueur* peut être indiquée par `i\Interligne` pour avoir *i* fois l'interligne. Par exemple,

```
\setinterinstrument 2{1\Interligne}
```

pour avoir les paroles entre le 2e et le 3e instrument.

Pour placer les paroles syllabe par syllabe, on utilise `\zsong{syllabe }` juste avant la note. Pour bien gérer l'empilement vertical, on metttra une seule syllabe par `\notes … \enotes`, par exemple

```
\NOtes \zsong{Au }\qu g\en
```

```
\NOtes \zsong{clair }\qu g\en
\NOtes \zsong{de }\qu g\en
\NOtes \zsong{la }\qu h\en
\bar
\NOtes \zsong{Lu- }\hu i\en
\NOtes \zsong{ne }\hu h\en
```

La commande `\zsong` place les paroles à la droite de la notre. On peut aussi utiliser `\csong` pour centrer, et `\lsong` pour les placer à gauche.

On peut ajuster la hauteur à laquelle est écrit le texte avec `\setsongraise` *n*`{`*longueur*`}` où *n* est le numéro de l'instrument concerné.

Pour régler les problèmes d'espacement, on peut utiliser les différents espacements (`\notes`, `\Notes`, `\NOtes`, …) ou bien faire calculer la taille du texte et réserver cette place pour la note avec :

```
\hardlyrics{texte}\notes \hsong{\thelyrics}note \en
```

où

- `\hardlyrics{`*texte*`}` calcule la largeur du *texte* et met celui-ci dans la variable `\thelyrics` ;
- il faut impérativement utiliser `\notes` et pas une autre commande ;
- `\hsong{…}` met le texte dans un boîte `\hbox`.

Placer du texte au-dessus des portées

Pour placer du texte au-dessus des portées, on utilise `\uptext`. Cela peut être un commentaire, mais aussi une indication comme `\textit{tr}` pour la trille.

Rappelons la commande `\znotes … \enotes` permet de ne pas provoquer d'espacement et donc de ne pas poser de problème d'alignement ; on peut alors faire

```
\znotes \uptext{texte} \enotes
```

Certains symboles sont placés en indiquant la note après ; par exemple, en clef de *sol*, pour placer le symbole au dessus de la portée, on peut utiliser la note `m` *(fa)* :

- `\segno m` pour le signe de répétition ;
- `\coda m` pour le ballon de coda.

Autres extensions

L'extension `gchords` permet de dessiner des diagrammes d'accord pour guitare.

Il est également possible d'utiliser gregorio (http://home.gna.org/gregorio/gregorio) [archive] qui est l'équivalent de lilypond pour le chant grégorien.

Voir aussi

- http://ctan.tug.org/tex-archive/macros/musixtex/taupin/

Références

1. le terme anglais est eighth note, mais MusiXTeX utilise l'acronyme français

Dessiner des échiquiers

Initialement, LaTeX disposait d'une police `chess` représentant les pièces d'échec ; le dessin de l'échiquier et le placement des pièces était à la charge de l'auteur du document. Puis est venu l'extension `skak`, amélioré par les fontes `SkakNew`, et enfin `chessboard`.

Par convention, sur le dessin, les blancs sont placés en bas au début du jeu. Les coordonnées des cases se composent

- de la colonne, qui est une lettre bas de casse (minuscule), de `a` pour la case la plus à gauche à `h` pour l'extrême droite ;
- et du numéro de ligne, de `1` pour la ligne du bas à `8` pour la ligne du haut.

Notations d'échecs

Aux échecs, on utilise plusieurs notations. Dans LaTeX, on en retient deux :

- la notation algébrique simplifiée anglaise ;
- la notation Forsyth-Edwards, ou FEN *(Forsyth-Edwards notation)*.

Voici quelques notions rapides.

Notation algébrique

La notation algébrique décrit les tours. Un tour se décompose en :

- le numéro du tour, suivi d'un point ;
- les deux mouvements, celui des blancs puis celui des noirs.

Si l'on veut juste indiquer le mouvement des noirs, on fait suivre le numéro de tour par trois points. Un mouvement se décompose en :

- de la nature de la pièce, par une lettre capitale (initiale du nom anglais) ou rien pour un pion ;
- des coordonnées de la case de départ ;
- d'un tiret « - » s'il s'agit d'un déplacement ou d'un « x » s'il s'agit d'une prise ;
- des coordonnées de la case d'arrivée.

Nature de la pièce

Nom français	Nom anglais	Abréviation
Roi	King	`K`
Dame	Queen	`Q`
Tour	Rook	`R`
Fou	Bishop	`B`
Cavalier	Knight	`N`
Pion	Pawn	*rien*

Par exemple

```
1. e2-e4 e7-e5
```

premier tour, le pion blanc situé en *e2* avance de deux case et vient en *e4* ; le pion noir qui lui fait face fait le même mouvement, il passe *e7* à *e5*

```
2. Ng1-f3 Nb8-c6
```

deuxième tour : le cavalier blanc en *g1* passe en *f3*, et le cavalier noir en *b8* passe en *c6*.

En notation abrégée, on n'indique pas la case de départ : la connaissance de la nature de la pièce et de la case d'arrivée suffit. Par exemple, le mouvement `Qd5xb7` (la reine en *d5* prend la pièce en *b7*) devient `Qxb7`.

L'exemple précédent en notation abrégée donne

```
1. e4 e5 2. Nf3 Nc6
```

Notation Forsyth-Edwards

La notation Forsyth-Edwards, ou FEN, permet de décrire un échiquier en place. Ceci est utile lorsque l'on est à une étape avancée d'une partie.

Cela consiste à noter ligne par ligne les pièces, et à indiquer le nombre de cases vides. La notation suit les règles suivantes :

- la description suit le sens de lecture : par ligne, on progresse de la case *a* à la case *h* (de la gauche vers la droite), en commençant par la ligne *8* (celle du haut) ;
- les lignes sont séparées par une barre de fraction « / » ;
- un chiffre *n* indique *n* cases vides ;
- les pièces son indiquées par une lettre, identique à la notation algébrique, mais le pion est noté *p* ; on utilise des lettres bas de casse pour les noirs, et capitales pour les blancs.

Par exemple, un échiquier au départ est décrit par :

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR
```

Puis, on indique

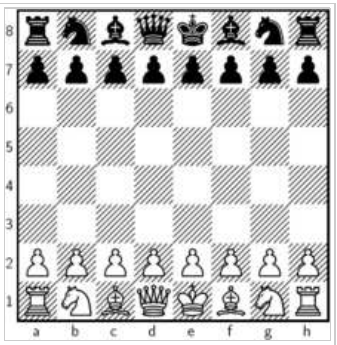
- qui joue le prochain tour : « w » pour les blancs *(white)* et « b » pour les noirs *(black)* ;
- les possibilités de roquer : un tiret « - » si ce n'est pas possible, sinon, la pièce pouvant roquer ;
- case d'une prise en passant, ou un tiret « - » s'il n'y en a pas ;
- le nombre de demi-coups depuis qu'un pion a été capturé ;
- le nombre de coups depuis le début de la partie.

La notation complète de l'échiquier de départ est donc

```
rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1
```

Avec skak

Skak ne se contente pas de l'affichage, il vérifie aussi la conformité des parties.



Échiquier obtenu avec skak

Comme toutes les extensions, il s'appelle dans le préambule par

```
\usepackage{skak}
```

La création d'un échiquier se passe en trois étapes :

- la commande `\newgame` définit un échiquier vide ;
- la commande `\mainline{`*`mouvements`*`}` définit les mouvements effectués en notation algébrique abrégée, affiche la ligne tapée et met à jour l'échiquier ;
- la commande `\showboard` affiche l'échiquier.

La commande `\variation{`*`mouvements`*`}` provoque l'affichage de mouvements mais pas la mise à jour de l'échiquier.

Si l'on veut que l'échiquier soit mis à jour mais que les mouvements ne s'affichent pas dans la lige descriptive, on utilise `\hidemove{`*`mouvements`*`}`. Cela peut permettre, par exemple, de montrer l'échiquier après quelques mouvements.

On peut utiliser la notation complète pour l'affichage en mettant la commande `\longmove` après le `\newgame` : on écrit en notation abrégée, mais `skak` affiche la notation complète. On peut revenir à la notation abrégée avec `\shortmove`. Par exemple, on écrit

```
\longmove
1. e4 c5
```

et le résultat est **1. e2-e4 c7-c5**.

Skak permet aussi d'utiliser la notation Forsyth-Edwards, ou FEN, avec la commande `\fenboard{…}`.

On peut faire varier la taille de l'échiquier :

```
\smallboard\showboard
\normalboard\showboard
\largeboard\showboard
```

Notons que `skak` permet d'enregistrer une partie en mémoire, avec la commande `\storegame{`*`nom de partie`*`}`. On peut la rappeler avec `\restoregame{`*`nom de partie`*`}`. Il peut même enregistrer la partie dans un fichier avec `\savegame{`*`nom de fichier`*`}`, et la recharger avec `\loadgame{`*`nom de partie`*`}`.

Avec chessboard

L'extension `chessboard` est une amélioration de `skak`. La syntaxe de `skak` est compatible avec `chessboard`.

Voir aussi

Généralités

- sur le site Enpassant.dk :
  - (anglais) Chess publishing (http://www.enpassant.dk/chess/dtpeng.htm) [archive]
  - (anglais) Chess fonts FAQ (http://www.enpassant.dk/chess/fonfaqen.htm) [archive]

CTAN

- Fonte Chess (http://ctan.tug.org/tex-archive/fonts/chess/chess/) [archive]
- Skak (http://ctan.tug.org/tex-archive/fonts/chess/skak/doc/) [archive]
- SkakNew (http://ctan.tug.org/tex-archive/fonts/chess/skaknew/) [archive]
- Chessboard (http://ctan.tug.org/tex-archive/macros/latex/contrib/chessboard/) [archive]

Wikipédia

- Jeu d'échecs

Table des matières - Généralités - Premiers pas - Structure du document - Gestion de la bibliographie - Tableaux - Images - Éléments flottants et figures - Mise en forme du texte - Choix de la police - Mise en page - Mathématiques - Gestion des gros documents - Faire des présentations - Arts et loisirs - Dessiner avec LaTeX - Créer une extension ou une classe - Programmer avec LaTeX - *Annexes* - Vade mecum - Conversion - Glossaire de typographie - Index - Commandes - Liens externes

Dessiner avec LaTeX

LaTeX propose des instructions pour faire des dessins, ou tracer des graphiques. Outre l'environnement `picture`, vous disposez des extensions de la suite PSTricks, qui permet également de tracer des fonctions mais doit être compilé avec `latex` et non pas avec `pdftex` (on peut obtenir un fichier PDF à partir du fichier PostScript généré), ou avec PGF/TikZ.

Il existe cependant des programmes de dessin qui génèrent du code PSTricks. On peut ainsi générer un fichier LaTeX et l'inclure dans le fichier en cours, pour générer une image de grande qualité pour une taille modeste (toutefois, le code généré n'est pas forcément très lisible ni optimal).

Citons par exemple TeXgraph de Patrick Fradin :

- page officielle TeXgraph (http://texgraph.tuxfamily.org/) [archive] ;
- forum TeXgraph (http://texgraph.heberg-forum.net/forums.html) [archive].

Bonnes habitudes de programmation

Séparation des graphiques du texte

Les dessins avec LaTeX sont des instructions LaTeX. Cependant, si ces instructions sont intégrées dans le texte, cela va produire un fichier source difficile à lire et donc à maintenir. Mis à part dans les cas simples, il vaut donc mieux mettre le code définissant l'image dans un fichier séparé et l'appeler à l'endroit voulu avec la commande `\input{`*`nom_de_fichier`*`}`.

Réutilisation d'objets

Un objet graphique peut être mis dans une commande personnelle (créée avec `\newcommand`) et être utilisé plusieurs fois.

On peut aussi mettre le texte de l'objet dans une boîte de sauvegarde et utiliser cette boîte, avec les commandes

1. `\newsavebox{\`*`nom`*`}` ;
2. `\sbox{\`*`nom`*`}{`*`objet`*`}` ou `\savebox{\`*`nom`*`}{`*`objet`*`}` ;
3. `\usebox{\`*`nom`*`}`,

où *nom* est le nom de la boîte.

L'avantage de cette méthode est que l'on peut indiquer la dimension de la boîte et le placement du texte au sein de la boîte avec `\savebox` :

```
\savebox{\nom}[largeur][position]{objet}
```

où *largeur* est un nombre avec unité, et *position* est `l`, `c` ou `r` pour aligné à gauche, centré ou aligné à droite. Dans l'environnement `picture`, `\savebox` peut prendre une autre syntaxe (voir la page suivante).

Note sur l'échelle

On peut dessiner en utilisant des valeurs exactes pour les dimensions. Cependant, dans certains cas, on peut avoir une altération de l'échelle dans la chaîne de création du document final.

Dans certains cas, ceci peut être réglé en modifiant le fichier `papersize` qui se trouve, sous Unix, dans `/etc/`. Par exemple, si l'on imprime au format A4, on mettra `A4` à la place de la valeur par défaut `letter`.

Sommaire

Dessiner avec LaTeX/Dessiner en LaTeX pur

LaTeX dispose en natif d'un certain nombre d'instructions de dessin. Toutefois, celles-ci s'appuient sur des caractères prédéfinis, les possibilités sont donc restreintes. De fait, peu de gens utilisent cette manière de faire. Cependant, pour des dessins simples, cela permet d'avoir un code universel, indépendant notamment du mode de compilation.

Les instructions spécifiques de dessin nécessitent d'être dans un environnement `picture`. Rappelons toutefois l'existence de la commande `\hrule` qui trace un trait de la largeur du paragraphe, et la commande `\rule` qui trace un trait de longueur et d'épaisseur déterminés[1] ; ces commandes sont utilisables hors environnement `picture`.

L'extension `pict2e` améliore de nombreuses choses à l'environnement `picture`, il est donc recommandé de le charger systématiquement.

Exemple complet et minimal

Le programme suivant affiche un trait barrant le texte.

```
\documentclass[10pt]{article}

\setlength{\unitlength}{1mm}

\begin{document}

\begin{picture}(0,0)
\line(5,1){5}
\end{picture}
Texte.

\end{document}
```

Le trait est tracé avec la commande `\line`. Le paramètre `{5}` indique que la longueur est 5 fois la longueur unité, définie par la commande `\setlength`, donc 5 mm. Le paramètre `(5,1)` indique que le trait est incliné dans des proportions 5/1.

La commande est encapsulée dans un environnement `picture`. Le paramètre `(0,0)` indique que l'on ne réserve pas de place à l'image : elle se superpose au texte.

On peut améliorer cet exemple :

```
\documentclass[10pt]{article}

\usepackage[latin1]{inputenc}
\usepackage[french]{babel}

\usepackage{pict2e}

\setlength{\unitlength}{1mm}

\begin{document}

\begin{picture}(0,0)
\line(10,1){20}
\end{picture}
Texte barré.

\end{document}
```

La première partie de l'entête est classique : elle permet d'utiliser les accents dans le code source (on peut écrire `barré` au lieu de `barr\'e`) et d'utiliser les règles de typographie française (voir *Premier exemple > Améliorations du code source*).

L'extension `pict2e` permet ici d'avoir un trait plus incliné : la proportion `(10,1)` n'est pas possible sans cette extension.

Définir l'unité par défaut et la largeur des traits

La première chose à faire est de définir l'unité par défaut, appelée `\unitlength`. Cela se fait avec l'instruction `\setlength` :

```
\setlength{\unitlength}{grandeur}
```

où *grandeur* est une longueur classique (un nombre assorti d'une unité), par exemple `1cm`, `1mm`, `1pt`, … Cette instruction se place de préférence

dans le préambule, mais on peut aussi la placer avant un graphique, pour en changer l'échelle (il ne faut alors pas oublier de restaurer la valeur par défaut après le graphique).

Par exemple

```
\setlength{\unitlength}{1mm}
```

On peut choisir d'avoir des lignes épaisses avec `\thicklines`, des lignes fines avec `\thinlines` (option par défaut), ou bien régler l'épaisseur avec

```
\linethickness{épaisseur}
```

où *épaisseur* est un nombre avec une dimension. Ces commandes peuvent se placer hors de l'environnement `picture`, ou à l'intérieur si des objets doivent avoir des épaisseurs de ligne différentes.

L'environnement `picture`

L'appel de l'environnement `picture` se fait en passant la dimension du graphique, exprimé en unité par défaut :

```
\begin{picture}(largeur,hauteur)
…
\end{picture}
```

Le graphique est alors positionné à l'emplacement du « curseur » (point à gauche sur la ligne d'écriture où serait placé le prochain caractère). Notons qu'il s'agit de la place qui est réservée au graphique sur la ligne (avant le positionnement du prochain caractère) ; si le dessin dépasse de ce cadre, il pourra se superposer au reste du texte.

On peut décaler le graphique par rapport au curseur ; LaTeX laisse toujours la même place réservée sur la ligne, mais le graphique est décalé :

```
\begin{picture}(largeur,hauteur)(x0,y0)
…
\end{picture}
```

où *x*0 et *y*0 sont les coordonnées du coin inférieur gauche en unité par défaut :

- un nombre négatif décale respectivement vers la gauche et vers le bas;
- un nombre positif décale respectivement vers la droite et vers le haut.

Notons que l'environnement `picture` peut être mis dans une figure flottante (environnement `figure`).

Placer un objet graphique

Le placement d'un objet dans le graphique se fait avec la commande `\put` :

```
\put(x,y){objet}
```

où *x* et *y* sont en unité par défaut. Le paramètre *objet* peut être du texte ou un dessin produit par une commande. Par exemple, le code suivant dessine un carré noir de 10 mm de côté avec marqué « carré » à 1 mm à droite :

```
\setlength{\unitlength}{1mm}
\begin{picture}(20,20)
\put(0,0){\rule{10mm}{10mm}}
\put(11,0){carré}
\end{picture}
```

Ici, la fonction `\rule` n'est pas une fonction de l'environnement `picture`, raison pour laquelle on doit lui indiquer les unités.

On peut insérer un objet plusieurs fois à intervalles réguliers. On utilise pour cela la fonction `\multiput` :

```
\multiput(x,y)(décalage_x,décalage_y){nombre_de_fois}{objet}
```

où

- *x* et *y* sont les coordonnées du premier objet ;
- *décalage_x* et *décalage_y* sont les espacement entre deux objets :
- *nombre_de_fois* est le nombre d'objets ainsi placés.

Par exemple, pour placer trois fois la lettre « a » sur une ligne :

```
\setlength{\unitlength}{1em}
\begin{picture}(4,2)
\multiput(0,0)(1,0){3}{a}
\end{picture}
```

Et pour faire un carré de 3×3 :

```
\setlength{\unitlength}{1em}
\begin{picture}(4,2)
\multiput(0,0)(1,0){3}{
   \multiput(0,0)(0,1){3}{a}
}
\end{picture}
```

Objets graphiques

Texte et cadre

Le texte peut simplement s'écrire tel quel. Toutefois, le texte est alors toujours sur une même ligne. Si l'on veut pouvoir utiliser le retour de ligne `\\`, il faut utiliser la commande `\shortstack` :

```
\setlength{\unitlength}{1mm}
\begin{picture}(40,20)
\put(0,0){
   \shortstack{texte sur\\
      plusieurs lignes}
}
\end{picture}
```

Le texte est alors centré. On peut passer l'alignement en paramètre : `l` pour aligner à gauche, et `r` pour aligner à droite, par exemple

```
\shortstack[l]{texte sur\\
   plusieurs lignes}
```

On peut encadrer le texte en utilisant la fonction `\framebox`. Par rapport à la fonction `\fbox`, elle utilise les unités par défaut et ne laisse pas de blanc autour. La syntaxe est

```
\framebox(largeur,hauteur){texte}
```

pour un texte centré dans le cadre, ou bien

```
\framebox(largeur,hauteur)[options]{texte}
```

où *options* indique l'alignement : `l` ou `r` pour aligner à gauche ou à droite, et `t` ou `b` pour aligner en haut ou en bas. On peut combiner deux alignements, par exemple

```
\framebox(10,10)[lt]{bla}
```

pour aligner en haut à gauche.

On peut aussi utiliser :

- `\makebox` pour avoir un cadre invisible ;
- `\dashbox` pour avoir un cadre en traits discontinus.

La fonction `\makebox` a la même syntaxe que `\framebox`. La syntaxe de `\dashbox` est :

```
\dashbox{longueur}(largeur,hauteur)[options]{texte}
```

où *longueur* est la longueur des tirets en unité par défaut.

Ces fonctions peuvent s'utiliser pour dessiner des rectangles, en mettant un texte vide.

La fonction `\put` place le coin en bas à gauche du rectangle.

Traits et flèches

Pour tracer un segment de droite, on utilise la fonction `\line` :

```
\line(directionx,directiony){longueur}
```

où *directionx* et *directiony* sont les coordonnées d'un vecteur directeur de la droite, et *longueur* est la longueur du segment, le tout exprimé en unité par défaut, avec les conventions suivantes :

- les coordonnées du vecteur directeur sont des nombres entiers ;
- pour les lignes horizontales ou verticales (dont l'une des composantes du vecteur directeur est nulle), la *longueur* est la longueur réelle du trait ;
- pour les lignes en biais (dont les deux composantes du vecteur directeur sont non nulles), la *longueur* est en fait la longueur de la projection sur l'axe des *x*.

Dans le cas d'une ligne en biais, la longueur réelle du trait est donc

$$\text{longueur réelle} = \text{longueur} \times \sqrt{1 + \frac{\text{direction } y^2}{\text{direction } x^2}}$$

Dans la pratique, si l'on veut tracer un trait de (0,0) à (*x*,*y*), *x* et *y* étant non nuls :

- si *x* et *y* sont entiers, on utilise `\line(x,y){x}` ;
- s'ils ne sont pas entiers, on utilise les valeurs multipliées par 10 ou par 100 : `\line(10*x,10*y){x}` (précision sur la direction de 0,1) ou bien `\line(100*x,100*y){x}` (précision sur la direction de 0,01) ;

la limite des valeurs avec `pict2e` étant 1 000 en valeur absolue. On peut aussi utiliser un facteur arbitraire, par exemple `\line(3*x,3*y){x}`.

Si l'on n'utilise pas `pict2e`, les valeurs sont limitées à 6 en valeur absolue et doivent être premières entre elles, ce qui limite à 25 les possibilités.

Une flèche se trace avec la commande `\vector` qui a la même syntaxe.

La fonction `\put` place l'extrémité de départ du segment ou du vecteur.

Par exemple

```
\setlength{\unitlength}{1mm}
\thicklines
\begin{picture}(40,20)
\put(0,0){
   \vector(1,1){10}
}
\end{picture}
```

Cercles, disques

On dessine un cercle avec la commande

```
\circle{diamètre}
```

où *diamètre* est dans l'unité par défaut. On dessine un disque avec

```
\circle*{diamètre}
```

La commande `\put` place le centre du cercle ou du disque.

Rectangles arrondis

On peut dessiner un rectangle aux coins arrondis avec

```
\oval(longueur,hauteur)
```

Si l'extension `pict2e` est chargée, on peut définir le rayon de courbure des coins :

```
\oval[rayon](longueur,hauteur)
```

On peut n'afficher qu'une moitié de boîte :

```
\oval(longueur,hauteur)[option]
```

ou *option* vaut `l` ou `r` pour la moitié gauche ou droite, `t` ou `b` pour la moitié haute ou basse. On peut n'afficher qu'un quart de la boîte en combinant deux options, par exemple

```
\oval(30,10)[lt]
```

pour le quart en haut à gauche. Les dimensions sont bien celles du rectangle complet.

La fonction `\put` place le centre du rectangle.

Courbes de Bézier

LaTeX permet de tracer un arc en utilisant une courbe de Bézier quadratique. Cette courbe est définie par trois points de contrôle A, B et C ; A et C sont les extrémités de la courbe, les tangentes aux extrémités passent par B.

La syntaxe est :

```
\qbezier(xA,yA)(xB,yB)(xC,yC)
```

Contrairement aux autres objets, cette commande ne se place pas dans une commande `\put`. On peut indiquer le nombre de points servant à tracer cette courbe ; alors, si ce nombre de points est suffisamment faible, on a une courbe pointillée :

```
\qbezier[nombre_de_points](xA,yA)(xB,yB)(xC,yC)
```

L'extension `pict2e` permet de tracer des courbes de Bézier cubiques :

```
\cbezier(xA,yA)(xB,yB)(xC,yC)(xD,yD)
```

Réutilisation d'objets

Comme indiqué précédemment, on peut créer des commandes personnelles (macro) pour réutiliser des objets. Mais on peut aussi utiliser des « boîtes de sauvegarde », qui permettent en outre de définir les dimensions de l'objet et le placement au sein de la boîte :

1. `\newsavebox{\nom}` : création de la boîte ;
2. `\savebox{\nom}{objet}` : définition du contenu de la boîte ;
3. `\usebox{\nom}` ; utilisation de la boîte,

où *nom* est le nom de la boîte.

Lorsque la commande `\savebox` est *à l'intérieur* d'un environnement `picture`, on peut utiliser la syntaxe suivante :

```
\savebox{\nom}(largeur,hauteur)[placement]{objet}
```

où (*largeur*,*hauteur*) est en unités par défaut, et *placement* prend les valeurs suivantes :

- `l`, `c`, `r` : centré verticalement, et respectivement collé à gauche, au centre et à droite ;
- `tl`, `tc`, `tr` : aligné en haut, et respectivement collé à gauche, au centre et à droite ;
- `bl`, `bc`, `br` : aligné en bas, et respectivement collé à gauche, au centre et à droite.

Extensions utiles

On peut utiliser l'extension `graphicx` qui permet de retailler (`\scalebox`), de faire tourner des objets (`\rotatebox`) et d'intégrer des images extérieures (`\includegraphicx`), voir *[[../../Mise en forme du texte (avancé)#Déformation du texte|Mise en forme du texte (avancé) > Déformation du texte]]*, *[[../../Inclure des images|Inclure des images]]* et *[[../../Images|Images]]*.

Par exemple, pour avoir une ellipse avec un rapport axe vertical sur horizontal de 1:2, on peut déformer un cercle :

```
\usepackage{graphicx}
…
\setlength{\unitlength}{1mm}
\begin{picture}(40,20)
\put(0,0){
   \scalebox{2}[1]{
      \circle{1}
   }
}
\end{picture}
```

cependant, le trait est plus large dans la dimension qui a été étirée.

L'extension `xcolor` permet de changer la couleur des objets dessinés. On utilise pour cela `\textcolor` (puisque les dessins sont à base de caractères). La commande peut être à l'intérieur de l'environnement `picture`, définissant ainsi la couleur de tel ou tel objet, ou bien peut contenir l'environnement `picture` et s'appliquer à tous les objets.

```
\usepackage{xcolor}
…
\setlength{\unitlength}{1mm}
\begin{picture}(40,20)
\textcolor{blue}{
   \put(0,0){
      \scalebox{2}[1]{
         \circle{1}
      }
   }
}
\end{picture}
…
\textcolor{blue}{
   \begin{picture}(40,20)
      …
   \end{picture}
}
```

L'extension `graphpap` permet d'obtenir du papier millimétré avec la commande `\graphpaper` :

```
\graphpaper[graduation](x,y)(largeur,hauteur)
```

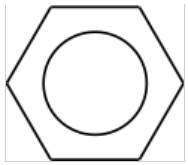
où *graduation* est l'espacement entre deux traits fins (par défaut 10), *x* et *y* sont l'origine des axes, *largeur* et *hauteur* sont les dimensions de la grille (en unité par défaut).

Exercices

Exercice 1

Réaliser la molécule de benzène ci-contre (format 2×2). On utilisera la géométrie pour avoir un placement exact des segments (les coordonnées des points et les directions sont calculées par l'utilisateur et entrées dans le code). Le rayon du cercle central est arbitraire et ajusté par essai-erreur « pour faire joli ».

Habituellement, on représente plutôt la groupement « debout » (tourné d'un sixième de tour par rapport au dessin présent). Pour le dessin des molécules, mieux vaut utiliser PPChTeX.



Exercice 1 — cliquer sur le lien « Dérouler »de la barre « Solution » ci-dessous pour avoir la solution

Indications

Il faut de placer le point (0,0) au centre du `picture`, les coordonnées du sommet *n* sont alors (cos(*n*·60°), sin(*n*·60°)) (*n* allant de 0 à 5).

Pour le placement d'un segment, on utilise les valeur décimales. Par contre, pour les directions, il faut multiplier les valeurs selon *x* et *y* par un même facteur et tronquer ; par exemple, un facteur 100 permet d'avoir une précision de 0,01 sur la direction, l'extension `pict2e` est nécessaire (sans `pict2e`, il faudrait approcher la direction (50, 87) par (3, 5)).

Solution

```
\documentclass{article}

\usepackage{pict2e}

\begin{document}

\setlength{\unitlength}{1cm}

\begin{picture}(2,2)(-1,-1)
   \put(1,0){\line(-50,87){0.5}}
   \put(0.5,0.87){\line(-1,0){1}}
   \put(-0.5,0.87){\line(-50,-87){0.5}}
   \put(-1,0){\line(50,-87){0.5}}
   \put(-0.5,-0.87){\line(1,0){1}}
```

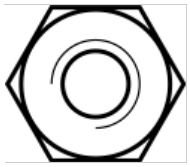
```
   \put(0.5,-0.87){\line(50,87){0.5}}
   \put(0,0){\circle{1.2}}
\end{picture}
```

Exercice 2

Réaliser l'écrou ci-contre (représentation de type dessin technique, format 2×2). Pour l'hexagone, on utilisera la solution ci-dessus.

Le trait fort a une épaisseur de 0,5 mm, soit `0.05` en unité par défaut ; le trait fin a une épaisseur de 0,2 mm, soit `0.02` en unité par défaut.

Notez que les dimensions ne correspondent pas à un écrou normalisé : le but n'étant pas de générer un dessin exact mais d'apprendre à maîtriser l'environnement `picture`, on utilise volontairement des valeurs simples.



Exercice 2 — cliquer sur le lien « Dérouler »de la barre « Solution » ci-dessous pour avoir la solution

Indications

Pour les trois quarts de cercle, on utilisera des rectangles aux angles arrondis dont le rayon de courbure est égale à la longueur des côtés. Il faut donc impérativement l'extension `pict2e`.

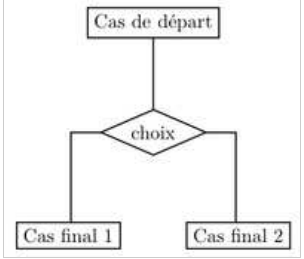
Solution

```
\documentclass{article}
\usepackage{pict2e}

\begin{document}

\setlength{\unitlength}{1cm}

\begin{picture}(2,2)(-1,-1)
   \linethickness{0.5mm}
   \put(1,0){\line(-50,87){0.5}}
   \put(0.5,0.87){\line(-1,0){1}}
   \put(-0.5,0.87){\line(-50,-87){0.5}}
   \put(-1,0){\line(50,-87){0.5}}
   \put(-0.5,-0.87){\line(1,0){1}}
   \put(0.5,-0.87){\line(50,87){0.5}}
   \put(0,0){\circle{1.74}}
   \put(0,0){\circle{0.8}}
   \linethickness{0.2mm}
   \put(0,0){\oval[0.5](1,1)[t]}
   \put(0,0){\oval[0.5](1,1)[br]}
\end{picture}

\end{document}
```

Exercice 3

Réaliser l'organigramme ci-contre (format 5×5).

L'ajustement des coordonnées et dimensions peut se faire par calcul et/ou essai-erreur.



Exercice 3 — cliquer sur le lien « Dérouler » de la barre « Solution » ci-dessous pour avoir la solution

Solution

```
\documentclass{article}
```

```
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
\usepackage{lmodern}
\usepackage{pict2e}

\begin{document}

\setlength{\unitlength}{1cm}

\begin{picture}(5,5)
%  traits entre les boîtes
   \put(2.5,3){\line(0,1){1}}
   \put(1,2.5){\line(1,0){0.5}}
   \put(3.5,2.5){\line(1,0){0.5}}
   \put(1,1.3){\line(0,1){1.2}}
   \put(4,1.3){\line(0,1){1.2}}

   \put(2.1,2.45){choix}

%  losange
   \put(2.5,3){\line(2,-1){1}}
   \put(2.5,3){\line(-2,-1){1}}
   \put(2.5,2){\line(-2,1){1}}
   \put(2.5,2){\line(2,1){1}}

   \put(1.25,4){\framebox(2.5,0.8)[c]{Cas de départ}}
   \put(0,0.5){\framebox(2.2,0.8)[c]{Cas final 1}}
   \put(2.8,0.5){\framebox(2.2,0.8)[c]{Cas final 2}}
\end{picture}

\end{document}
```

Exercice 4

Réaliser le document ci-contre.

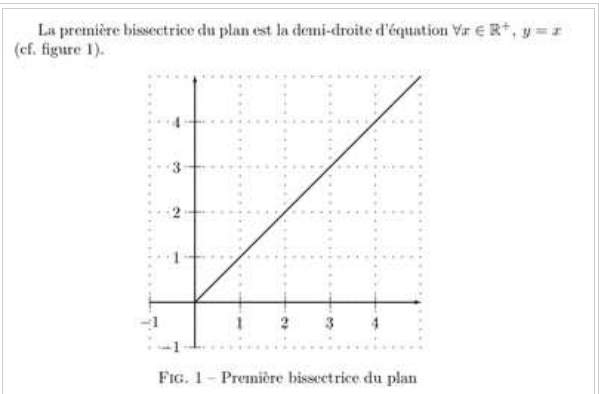


FIG. 1 – Première bissectrice du plan

Exercice 4 — cliquer sur le lien « Dérouler » de la barre « Solution » ci-dessous pour avoir la solution

Solution

Fichier `exercice.tex`

```
\documentclass{article}

\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
\usepackage{amsfonts} % pour \mathbb (pour l'ensemble des réels)
\usepackage[frenchb]{babel}

\begin{document}
La première bissectrice du plan
est la droite d'équation
$\forall x \in \mathbb{R}^+,\ y = x$
(cf. figure~\ref{premierebissectrice}).
\begin{figure}[h]
   \centering
   \input{image1}
   \caption{\label{premierebissectrice}Première bissectrice du plan}
\end{figure}
\end{document}
```

Fichier `image1.tex`

```
\setlength{\unitlength}{1cm}
\begin{picture}(6,6)(-1,-1)
```

```
% axe x
   \put(0,-1){\vector(0,1){6}}
   \multiput(-1,-0.2)(1,0){6}{\line(0,1){0.4}} % traits de graduation
   \put(-1.2,-0.6){$-1$}
   \put(0.9,-0.6){1} % valeurs graduées
   \put(1.9,-0.6){2}
   \put(2.9,-0.6){3}
   \put(3.9,-0.6){4}
% axe y
   \put(-1,0){\vector(1,0){6}}
   \multiput(-0.2,-1)(0,1){6}{\line(1,0){0.4}} % traits de graduation
   \put(-0.8,-1.1){$-1$} % valeurs graduées
   \put(-0.5,0.9){1}
   \put(-0.5,1.9){2}
   \put(-0.5,2.9){3}
   \put(-0.5,3.9){4}
% quadrillage
   \multiput(-1,-1)(1,0){7}{
      \multiput(0,0)(0,0.2){31}{\circle*{0.05}}}
   \multiput(-1,-1)(0,1){7}{
      \multiput(0,0)(0.2,0){31}{\circle*{0.05}}}
% demi-droite
   \put(0,0){\line(1,1){5}}
\end{picture}
```

On peut faire un code un peu plus compact en ajoutant

```
\usepackage{multido}
```

dans l'en-tête du fichier `exercice.tex`, puis dans le fichier `image1.tex` :

```
…
% axe x
   \put(0,-1){\vector(0,1){6}}
   \multiput(-1,-0.2)(1,0){6}{\line(0,1){0.4}} % traits de graduation
   \put(-1.2,-0.6){$-1$}  % valeurs graduées
   \multido{\i=1+1}{4}{\put(\i,-0.6){\put(-0.1,0){\i}}}
% axe y
   \put(-1,0){\vector(1,0){6}}
   \multiput(-0.2,-1)(0,1){6}{\line(1,0){0.4}} % traits de graduation
   \put(-0.8,-1.1){$-1$}  % valeurs graduées
   \multido{\i=1+1}{4}{\put(-0.5,\i){\put(0,-0.1){\i}}}
…
```

L'imbrication `\put(\i,-0.6){\put(-0.1,0){\i}}` sert à avoir l'équivalent de `\i-0.1` pour le placement en *x* (l'extension `calc` pose problème avec l'unité par défaut).

Notes

1. la syntaxe est
   `\rule{`*`longueur`*`}{`*`épaisseur`*`}`
   on utilise parfois cette commande pour « réserver de la place » en traçant un trait de longueur nulle ou d'épaisseur nulle

Voir aussi

Sur Wikipédia

- Courbe de Bézier

Dessiner avec LaTeX/Dessiner avec PSTricks

PSTricks est une distribution de plusieurs extensions. L'extension de base est `pstricks`, d'autres extensions peuvent être chargées selon les besoins. L'extension `pstricks` charge automatiquement l'extension `xcolor`[1], la couleur est donc automatiquement prise en compte.

L'utilisation de PSTricks nécessite de passer par du PostScript, on ne peut donc pas compiler avec `pdftex`/`pdflatex`. On peut toutefois obtenir un fichier PDF à partir du fichier PostScript en utilisant, par exemple, `ps2pdf`. Signalons que l'extension PDFtricks permet de placer directement des commandes PSTricks dans un document écrit pour pdflatex (voir ci-après).

Les unités par défaut sont le centimètre pour les longueurs et le degré pour les angles. On peut toutefois utiliser des longueurs avec unité.

Contrairement aux commandes de `picture`, les commandes de PSTricks peuvent être utilisées en dehors d'un environnement graphique, ce sont alors des objets de « largeur nulle » (le caractère suivant est écrit comme s'il n'y avait pas de dessin). Par exemple, si l'on veut rayer un mot, on peut créer la commande suivante :

```
\usepackage{pstricks}

\newlength{\longueurmot}
\newcommand{\rayer}[1]{%
\settowidth{\longueurmot}{#1}%
#1\psline(0,0.5ex)(-\longueurmot,0.5ex)%
}
```

(voir aussi [[../../Mise en forme du texte (avancé)#textbarre1|ici]] une autre solution).

Environnement `pspicture`

Si l'on veut que LaTeX prenne en compte le graphique pour placer le texte, il faut placer les instructions dans un environnement `pspicture`. On passe en paramètre les coordonnées du point en haut à droite, le point (0,0) étant au point de référence (sur la ligne de base là où serait le prochain caractère) :

```
\begin{pspicture}(x1,y1)
   …
\end{pspicture}
```

on peut aussi indiquer les coordonnées du point en bas à gauche :

```
\begin{pspicture}(x0,y0)(x1,y1)
   …
\end{pspicture}
```

cela décale la position du point (0,0), mais ne change pas l'emplacement de la boîte. Les dimensions de la boîte sont $(x_1\text{-}x_0)\times(y_1\text{-}y_0)$.

On peut mettre l'environnement `pspicture` dans une figure flottante (environnement `figure`).

Objets de base

Segments et lignes brisées

Un segment de droite s'obtiennent avec

```
\psline(x0,y0)(x1,y1)
```

Pour tracer un vecteur[2] :

```
\psline{->}(x0,y0)(x1,y1)
```

Une ligne brisée s'obtient en mettant les points les uns à la suite des autres

```
\psline(x0,y0)(x1,y1)(x2,y3)…(xn,yn)
```

On peut tracer une ligne brisée aux angles arrondis avec l'option `linearc=valeur`, la *valeur* en question étant le rayon de courbure.

Exemple

```
\psline[linearc=0.2,->](0,0)(0.5,0.5)(1,1)
```

Rectangles

Pour tracer un rectangle, on utilise la commande `\psframe`, en indiquant les coordonnées des deux coins en bas à gauche et en haut à droite :

```
\psframe(x0,y0)(x1,y1)
```

La commande `\psframe*` a la même syntaxe donne un rectangle rempli.

On peut tracer des rectangles aux angles arrondis avec l'option `framearc=valeur`, la *valeur* en question étant le rayon de courbure.

Exemple

```
\psframe[framearc=0.2](0,0)(1,1)
```

Polygones

Les polygones sont toujours fermés. Ils sont décrits avec la commande `\pspolygon` et ont la même syntaxe que `\psline` :

```
\pspolygon(x0,y0)(x1,y1)(x2,y3)…(xn,yn)
```

La commande `\pspolygon*` donne un polygone rempli.

On peut tracer des polygones aux angles arrondis avec l'option `linearc=valeur`, la *valeur* en question étant le rayon de courbure.

Exemple

```
\pspolygon[linearc=0.2](0,0)(0.5,0.5)(1,1)
```

Cercles, arcs de cercle, portions de disque et ellipses

Les cercles se tracent avec la commande `\pscircle` :

```
\pscircle(x,y){r}
```

où (*x*,*y*) sont les coordonnées du cercle et *r* est son rayon. La commande `\pscircle*` donne un disque plein.

On peut tracer un arc de cercle avec la commande `\psarc` :

```
\psarc(x,y){r}{angle1}{angle2}
```

où *angle*1 et *angle*2 sont les angles limitant de l'arc, selon la convention trigonométrique. La commande `\psarc*` donne une portion de disque pleine délimitée par l'arc et la corde.

On peut tracer une portion de disque façon « part de camembert » avec la commande `\pswedge`, qui a une syntaxe similaire à `\psarc`. La commande `\pswedge*` trace une portion pleine.

Les ellipses se tracent avec la commande `\psellipse` :

```
\psellipse(x,y)(axe_horizontal,axe_vertical)
```

où (*axe_horizontal*,*axe_vertical*) sont les longueurs des axes. La commande `\psellipse*` donne une ellipse pleine.

Exemple

```
\psarc[->](0,0){1}{270}{315}
```

Courbes

Paraboles

On peut tracer un arc de parabole de direction asymptotique verticale avec la commande `\psparabola` :

```
\psparabola(x0,y0)(x1,y1)
```

le somme de la parabole est en (*x*1,*y*1), elle passe par (*x*0,*y*0) et s'arrête à ce point, et les deux branches sont symétriques.

La commande `\psparabola*` donne un portion de plan pleine délimitée par l'arc et la corde.

Courbes de Bézier

La commande `\psbezier` permet de tracer une courbe de Bézier avec un nombre arbitraire de points de contrôle, mais composée d'arcs ayant au plus quatre points de contrôle. L'option `showpoints=true` permet d'afficher les points de contrôle ainsi que les tangentes. La commande `\psbezier*` donne une surface pleine délimitée par la courbe et la droite joignant les points extrêmes.

Courbe d'interpolation

La commande `\pscurve` crée une courbe passant par tous les points indiqués. La commande `\psecurve` ne trace pas les premier et dernier arcs, mais prend en compte les points extrêmes pour le calcul de la courbure ; cela permet de s'affranchir d'effets de bord en « forçant » la courbure ou la tangente aux extrémités tracées.

Les commandes étoilées `\pscurve*` et `\psecurve*` tracent des surfaces pleines délimitées par la courbe et la droite joignant les points extrêmes.

Texte

On peut écrire du texte à un point donné avec la commande `rput` :

```
\rput(x,y){texte}
```

On peut faire tourner le texte d'un angle donné :

```
\rput{angle}(x,y){texte}
```

On peut passer en option le point de la boîte de texte qui sera au point (*x*,*y*) désigné :

- `B`, `Bl` et `Br` : respectivement le centre, l'extrême gauche et l'extrême droite de la ligne de base d'écriture *(base)* ;
- `t`, `tl` et `tr` : respectivement le centre, l'extrême gauche et l'extrême droite du haut de la boîte *(top)* ;
- `b`, `bl` et `br` : respectivement le centre, l'extrême gauche et l'extrême droite du bas de la boîte *(bottom)*

```
\rput[référence](x,y){texte}
```

Exemple

```
\rput[t]{45}(5,5){texte}
```

On peut aussi utiliser la commande `\uput`, qui elle place le texte en décalé par rapport au point indiqué :

```
\uput{distance}[direction](x,y){texte}
```

va placer le texte à une distance donnée dans une direction donnée (définie par un angle) par rapport à (*x*,*y*). On peut aussi faire tourner le texte d'un angle donné :

```
\uput{distance}[direction]{rotation}(x,y){texte}
```

PSTricks propose plusieurs encadrements pour le texte :

- `\psframebox{texte}` : cadre rectangulaire ;
- `\psdblframebox{texte}` : cadre rectangulaire à filet double ;
- `\psshadowbox{texte}` : cadre rectangulaire ombré ;
- `\pstcirclebox{texte}` : cadre circulaire ;
- `\psovalbox{texte}` : cadre oval ;

- `\psdiabox{texte}` : cadre losange ;
- `\pstribox{texte}` : cadre triangulaire (triangle isocèle pointe en haut).

Les versions étoilées donnent un cadre plein.

Exemple

```
\rput(5,5){\psdiabox*[fillcolor=green]{texte}}
```

L'extension `pst-text` permet de faire en sorte que le texte épouse une ligne, par exemple obtenue avec `\psline`, `\pspolygon`, `\pscurve`, … On utilise pour cela la commande `\pstextpath` :

```
\pstextpath{ligne}{texte}
```

Mais il faut auparavant annuler le tracé des lignes avec `\psset{linestyle=none}` si l'on veut avoir le texte seul.

Exemple

```
\usepackage{pst-text}
…
\begin{pspicture}(5,5)
\psset{linestyle=none}
\pstextpath{\psline(0,0)(1,1)(2,0)}{texte en triangle}
\end{pspicture}
```

Quadrillage et axes

La commande `\psgrid` crée un quadrillage sur toute l'image (dans un environnement `pspicture`), avec un pas de 0,2 (2 mm par défaut). On peut lui adjoindre des paramètres :

- `\psgrid(xmax,ymax)` : fait un quadrillage de (0,0) à (*xmax*,*ymax*) ;
- `\psgrid(xmin,ymin)(xmax,ymax)` : fait un quadrillage de (*xmin*,*ymin*) à (*xmax*,*ymax*) ;
- `\psgrid(x0,y0)(xmin,ymin)(xmax,ymax)` : comme précédemment, et le un des nœuds du quadrillage est en (*x*0,*y*0).

On peut utiliser les options suivantes :

- `griddots=valeur` : le trait plein des graduations principales est remplacé par un trait pointillé, on indique le nombre de points par graduation ;
- `subgriddots=valeur` : idem avec les sous-gradations ;
- `gridcolor=couleur`, `subgridcolor=couleur` : couleur des traits de graduation et sous-graduation ;
- `gridwidth=valeur`, `subgridwidth=valeur` : épasseur des traits de graduation et sous-graduation ;
- `subgriddiv=valeur` : nombre de sous-graduation entre deux graduations principales ;
- `gridlabels=valeur` : taille des nombres étiquetant les graduations ;
- `ticksize=valeur` : taille des graduations ;
  - `ticksize=valeur valeur` si l'on indique deux valeurs, on a des graduations dissymétriques : la première valeur est la longueur du côté des coordonnées négatives, la deuxième est celle du côté des coordonnées positives ;
- `ticklinestyle=valeur` : style des graduations (`solid`, `dashed`, `dotted`) ; utile lorsque les graduations sont très grandes (`ticksize` élevé), ce qui permet de faire un quadrillage (les case de `\psgrid` dépendent uniquement de l'unité).

Exemple

```
\psgrid[griddots=5, subgriddiv=0, gridlabels=0pt](-1,-1)(5,5)
```

Si l'on veut placer des axes, il faut utiliser l'extension `pstricks-add` qui fournit la commande `\psaxes` :

```
\psaxes(xmin,ymin)(xmax,ymax)
\psaxes(x0,y0)(xmin,ymin)(xmax,ymax)
```

où (*xmin*,*ymin*) et (*xmax*,*ymax*) sont les extrêmes des axes, et (*x*0,*y*0) est le point où se coupent les axes.

On peut ajouter des options :

- `Dx=valeur` et `Dy=valeur` permettent de définir l'espacement entre les graduations ;
- `comma` permet d'utiliser la virgule comme séparateur décimal.

Par ailleurs, comme pour les segments, `{->}` permet de mettre des flèches aux axes.

Exemple

```
\usepackage{pstricks-add}
…
\begin{pspicture}(-1,-1)(5,5)
\psaxes[comma,Dx=0.5,Dy=0.5]{->}(0,0)(3,3)
\end{pspicture}
```

Paramètres généraux

Toutes les figures

On peut ajouter des paramètres optionnels de traits, entre crochet :

- `linewidth=`*`valeur`* : épaisseur du trait, *valeur* étant un nombre seul (unité par défaut) ou un nombre avec unité ;
- `linecolor=`*`couleur`* : couleur du trait (la couleur est définie avec les commandes de `xcolor`) ;
- `linestyle=`*`valeur`* : style de ligne, *valeur* pouvant être `dashed` (tirets), `dotted` (pointillés) ;
- `doubleline=true` : trait double ;
- `showpoints=true` : les extrémités des segments sont surlignés ; on peut définir la taille des points avec `dotscale=`*`valeur`*, et le type de points avec `dotsyle=`*`valeur`*, les valeurs pouvant être
  - `*` : disque ;
  - `o` : cercle ;
  - `+`, `x` : croix ;
  - `square`, `square*` : carré vide ou plein ;
  - `diamond`, `diamond*` : losange vide ou plein ;
  - `triangle`, `triangle*` : triangle vide ou plein ;
  - …

Par exemple

```
\pscircle[linewidth=0.2,linestyle=dashed,linecolor=blue](0,0){1}
```

Si des paramètres s'appliquent à plusieurs figures, on peut les définir avec la commande `\psset`, par exemple

```
\psset{linewidth=0.2,linestyle=dashed,linecolor=blue}
\pscircle(0,0){1}
```

Cette commande permet également de changer l'unité par défaut avec les paramètres :

- `unit=`*`valeur`*, ou
- `xunit=`*`valeur`* et `yunit=`*`valeur`* si l'on veut avoir des unités différentes selon l'axe,

valeur étant un nombre avec ou sans unité. Cela permet de changer l'échelle d'un dessin, de bénéficier de la graduation automatique des axes, et ne modifie pas l'épaisseur des traits.

Figures ouvertes

On peut définir l'extrémité d'une figure ouverte (segment, ligne brisée, arc, …) avec un paramètre optionnel, sous la forme :

```
\psline{''extrémité0''-''extrémité1''}(''x''0,''y''0)(''x''1,''y''1)
```

Les différentes extrémités sont :

- `<` ou `>` : flèche ;
- `<<` ou `>>` : flèche double ;
- `|` : barre collée à l'extrémité ;
- `|*` : barre centrée sur l'extrémité ;
- `o` : cercle centré sur l'extrémité ;
- `oo` : cercle collé à l'extrémité ;
- `*` : disque centré sur l'extrémité ;
- `**` : disque collé à l'extrémité ;
- `|<` et `>|` : flèche centrée sur l'extrémité et barre ;
- `c` : extrémité arrondie, le disque étant centré sur l'extrémité ;
- `cc` : extrémité arrondie, le bord du disque étant à l'extrémité.

Par exemple : `<->`, `>-<`, `|->`, `|<->|`.

Figures fermées

Pour les figures fermées, on peut définir le type de remplissage :

- `fillstyle=`*`valeur`* : motif de remplissage, les valeurs pouvant être :
  - `crosshatch` : hachures croisées à 45 °, quadrillage transparent,
  - `crosshatch*` : *idem*, quadrillage opaque,
  - `vlines` et `vlines*` : hachures simples à -45 °, respectivement transparent et opaque,
  - `hlines` et `hlines*` : hachures simples à 45 °, respectivement transparent et opaque,
  - `solid` : plein ;
- `fillcolor=`*`couleur`* : couleur de fond ;
- `hatchcolor=`*`couleur`* : couleur des hachures ;
- `hatchwidth=`*`valeur`* : épaisseur du trait ;
- `hatchsep=`*`valeur`* : espacement des traits ;
- `hatchangle=`*`valeur`* : angle des traits.

Par exemple

```
\pscircle[hatchcolor=blue,fillstyle=vlines](0,0){1}
```

Placement des objets

Les commandes `\rput` et `\uput` peuvent en fait servir à translater n'importe quel objet.

Exemple

```
\begin{pspicture}(5,5)
\psline{->}(0,0)(1,1)
\rput(1,1){\psline{->}(0,0)(1,1)}
\end{pspicture}
```

est équivalent à

```
\begin{pspicture}(5,5)
\psline{->}(0,0)(1,1)
\psline{->}(1,1)(2,2)
\end{pspicture}
```

On peut répéter le placement avec la commande `\multirput` :

```
\multirput(''x''0,''y''0)(''décalage_x'',''décalage_y''){''nombre_de_fois''}{''objet''}
```

où (*décalage_x*,*décalage_y*) est le vecteur séparant deux objets placés consécutivement. On peut ajouter les même options qu'avec `\rput` (point de référence et angle de rotation) :

```
\multirput[''référence'']{''angle''}(''x''0,''y''0)(''décalage_x'',''décalage_y''){''nombre_de_fois''}{''objet''}
```

S'il n'y a pas de texte mais uniquement des objets graphiques, on peut utiliser la commande `\multips` :

```
\multips(''x''0,''y''0)(''décalage_x'',''décalage_y''){''nombre_de_fois''}{''objet''}
\multips{''angle''}(''x''0,''y''0)(''décalage_x'',''décalage_y''){''nombre_de_fois''}{''objet''}
```

Extension PDFTricks

L'extension PDFTricks permet d'utiliser PSTricks avec `pdflatex`. Pour cela, il faut

- déclarer l'extension PDFTricks dans l'entête ;
- placer les extensions PSTricks dans une environnement `psinputs`, et intégrer toutes les commandes PSTricks dans un environnement `pdfpic` ;
- sous les environnements Unix, autoriser le lancement d'une commande en ligne depuis le compilateur `pdflatex`.

Le fichier `.tex` ressemblera à

```
\documentclass{article}

\usepackage{pdftricks}
\begin{psinputs}
   \usepackage{pstricks}
   \usepackage{multido}
\end{psinputs}

[...]
```

```
\begin{document}

[...]

\begin{pdfpic}
   \psset{unit=\linewidth}
   \begin{pspicture}(0,0)(10,10)
      [...]
   \end{pspicture}
\end{pdfpic}

[...]

\end{document}
```

Pour autoriser le lancement d'une commande en ligne, on lance la compilation avec l'option `-shell-escape` :

```
pdflatex -shell-escape {nom_du_fichier}
```

Exercices

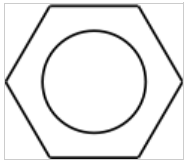
Dans les exercices suivants, le but est de :

- créer des fichiers compilables, notamment contenant un `\begin{document}` et un `\end{document}` ; la classe importe peu ;
- en séparant le fichier image du fichier principal si le document mèle texte et images..

Exercice 1

Réaliser la molécule de benzène ci-contre (format 2×2). On utilisera la géométrie pour avoir un placement exact des segments. On donnera deux solutions :

1. les coordonnées des points sont calculées par l'utilisateur et entrées dans le code ;
2. on ne calcule les coordonnées que pour deux sommets consécutifs, puis on utilise une rotation pour placer les autres ; cette solution donne des effets étranges lorsque l'épasseur du trait devient importante et n'est donc pas recommandée, mais est présentée ici à titre d'exercice.



Exercice 1 — cliquer sur l'image pour avoir le code

Le rayon du cercle central est arbitraire et ajusté par essai-erreur « pour faire joli ».

Habituellement, on représente plutôt la groupement « debout » (tourné d'un sixième de tour par rapport au dessin présent). Pour le dessin des molécules, mieux vaut utiliser PPChTeX.

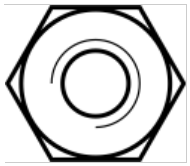
Indications

Pour la première solution, il suffit de placer le point (0,0) au centre du `pspicture`, les coordonnées du sommet n sont alors (cos(n·60°), sin(n·60°)) (n allant de 0 à 5).

Pour la seconde solution, l'idée est de partir d'un segment décalé par rapport au centre de l'image, et de faire tourner ce segment de 60° à chaque fois. Il faut donc que le point (0,0) soit au centre du `pspicture`. On pourra par exemple prendre le segment du bas comme référence : ses points sont (-sin(30°), -cos(30°))–((sin(30°), -cos(30°)), c'est en fait la base d'un triangle équilatéral dont le sommet est en (0,0).

Exercice 2

Réaliser l'écrou ci-contre (représentation de type dessin technique, format 2×2). Pour l'hexagone, on utilisera la première solution évoquée ci-dessus.



Exercice 2 — cliquer sur l'image pour avoir le code

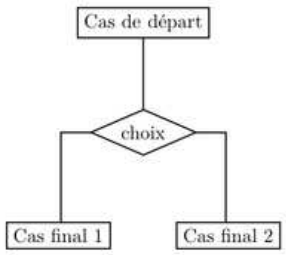
Le trait fort a une épaisseur de 0,5 mm, soit `0.05` en unité par défaut ; le trait fin a une épaisseur de 0,2 mm, soit `0.02` en unité par défaut.

Notez que les dimensions ne correspondent pas à un écrou normalisé : le but n'étant pas de générer un dessin exact mais d'apprendre à maîtriser PSTricks, on utilise volonteairement des valeurs simples.

Exercice 3

Réaliser l'organigramme ci-contre (format 5×5).

L'extension `pst-node` permet de simplifier la réalisation de ce type de diagrammes, comme on le verra plus loin.



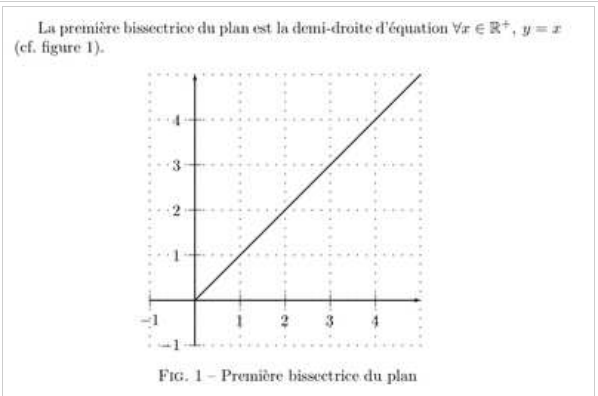
Exercice 3 — cliquer sur l'image pour avoir le code

Indications

pour éviter d'avoir à ajuster la longueur des traits, on ne cherchera pas à s'arrêter exactement au bord des cadres, et on affichera les boîtes de texte en dernier en utilisant l'option `fillstyle=solid`. On peut utiliser astucieusement les options de placement par rapport au point de référence.

Exercice 4

Réaliser le document ci-contre.



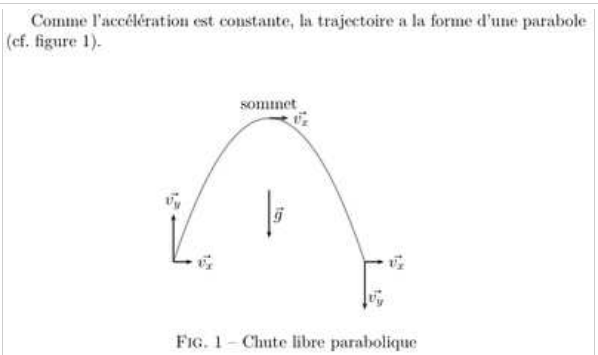
Exercice 4 — cliquer sur l'image pour avoir le code

Exercice 5

Réaliser le document ci-contre, en créant des commandes personnelles pour les objets présents plusieurs fois.

L'image fait 5×5 (unité par défaut), on laisse une marge de 1 en haut, en bas et à droite de la parabole pour pouvoir placer les indications. La tangente aux extrémités faisant un angle supérieur à 45 ° par rapport à l'horizontale, le vecteur v_x est plus petit que le vecteur v_y , mais la valeur exacte importe peu pour cette représentation schématique.

On placera les éléments de texte relativement aux points particuliers du dessin (extrémités ou milieu des vecteurs, sommet de la parabole).



Exercice 5 — cliquer sur l'image pour avoir le code

Extensions complémentaires

Gestion des nœuds

Représentation d'arborescences

Notes

1. l'extension `pstcol` est obsolète
2. contrairement à l'habitude, certains paramètres optionnels sont entre accolades dans PSTricks, c'est le cas ici de `->`

Voir aussi

Sur Wikipédia

- Courbe de Bézier
- PSTricks

sur CTAN

- documentation de l'extension (http://ctan.tug.org/tex-archive/graphics/pstricks/base/doc/) [archive]

Dessiner avec LaTeX/Dessiner avec PGF et TikZ

PGF signifie *« portable graphics format »*, ou *« pretty, good, functional »*. Contrairement à ce qu'indique son nom, ce n'est pas un format au sens « format de fichier », mais un ensemble d'instructions pour (La)TeX permettant de faire des dessins vectoriels, et la portabilité est limitée au monde de (La)TeX (TeX, LaTeX, ConTeXt), pour faire du PS ou du PDF. Les instructions PGF se placent au sein d'un environnement `pfgpicture`. Cela permet de générer du PostScript et du PDF.

Ti*k*Z est une extension permettant de générer des images PGF avec une syntaxe assez simple ; c'est en quelques sorte une « surcouche » de PGF pour LaTeX, Ti*k*Z est à PGF ce que PSTricks est au PostScript. Selon son auteur Till Tantau, Ti*k*Z est un acronyme récursif qui veut dire « Ti*k*Z ist kein Zeichenprogramm » : Ti*k*Z n'est *pas* un logiciel de dessin - au sens de « Ti*k*Z est un langage pour graphiques ».

Environnement `tikzpicture`

L'environnement `tikzpicture` permet de déclarer à LaTeX que l'on commence une image Ti*k*Z. On peut définir un facteur d'échelle ; par exemple, si l'on veut multiplier la taille de l'image dessinée par trois, on utilise :

```
\begin{tikzpicture}[scale=3]
   ...
\end{tikzpicture}
```

Pour les petites images, on peut se contenter d'utiliser la commande `\tikz` : cette commande prend l'argument qui suit, typiquement une commande de dessin et ses paramètres, et l'inclut dans un environnement `tikzpicture`. On peut soit mettre la commande et ses paramètres entre accolades, soit terminer la commande par un point virgule :

```
\tikz{ \draw (0,0) -- (1,1)}
```

et

```
\tikz \draw (0,0) -- (1,1) ;
```

équivalent à

```
\begin{tikzpicture}
   \draw (0,0) -- (1,1) ;
\end{tikzpicture}
```

Coordonnées

L'unité de longueur par défaut est le centimètre ; l'unité d'angle est le degré. On peut définir un point par ses coordonnées cartésiennes

```
(x,y)
```

ou par ses coordonnées polaires

```
(angle:rayon)
```

Par défaut, pour simplifier la présentation, nous considèrerons des coordonnées cartésiennes, mais les coordonnées polaires peuvent être utilisées partout.

Chemin

Le chemin *(path)* est la notion de base de Ti*k*Z. Un chemin est une série de points séparés par des « opérations d'extension ». Les opérations d'extensions définissent le type de trait qui relient les points :

- `--` (deux tirets) : un segment de droite relie les deux points ;
- `rectangle` : trace un rectangle horizontal dont les angles sont les deux points.

Par exemple, un chemin sous la forme d'une ligne brisée sera de la forme

```
(x0,y0) -- (x1,y1) -- (x2,y2) -- (x3,y3)
```

Si l'on termine le chemin par `-- cycle`, il sera fermé.

Les points peuvent être définis de manière relatives par rapport à un point précédent : le signe + devant des coordonnées définit un décalage, mais laisse inchangé le point de référence (le point de départ reste la référence), tandis que l'opérateur ++ considère que le point ainsi défini est la nouvelle référence pour les points suivants.

Par exemple,

```
(0,0) -- + (0,1) -- + (1,0)
```

est équivalent à

```
(0,0) -- (0,1) -- (1,0)
```

tandis que

```
(0,0) -- ++ (0,1) -- ++ (1,0)
```

est équivalent à

```
(0,0) -- (0,1) -- (1,1)
```

Un cercle est un chemin défini à partir de son centre et de son rayon :

```
(x0,y0) circle (rayon)
```

où *rayon* est une longueur indiquée comme à l'habitude sous la forme nombre et unité. par exemple, le cercle de centre O et de rayon 1 cm est défini par :

```
(0,0) circle (1cm)
```

Pour avoir un arc de cercle, on utilise

```
(x0,y0) arc (angledébut:anglefin:rayon)
```

par exemple, pour avoir l'arc du cercle précédent compris entre 0 et 90 °, on écrit :

```
(0,0) arc (0:90:1cm)
```

Une ellipse est un chemin défini à partir de son centre et de ses axes horizontal et vertical : :

```
(x0,y0) ellipse (axehorizontal and axevertical)
```

Pour les courbes de Bézier, les points de passage sont séparés par les points de contrôle :

```
(x0,y0) .. controls (x1,y1) .. (x2,y2)
```

est l'arc dont les extrémités sont (x_0, y_0) et (x_2, y_2), et dont le point de contrôle est (x_1, y_1). S'il y a plusieurs points de contrôle, ils sont séparés par `and`. Par exemple, avec deux points de contrôle :

```
(x0,y0) .. controls (x1,y1) and (x2,y2) .. (x3,y3)
```

Traçage de ligne

La commande permettant de tracer une ligne est

```
\draw chemin
```

où chemin est défini comme précédemment. Par exemple, pour tracer le cercle de centre O et de rayon 1 cm, on écrit :

```
\draw (0,0) circle (1cm);
```

On peut choisir la couleur du tracé :

```
\draw[red] (0,0) -- (1,1);
```

On peut aussi ajouter des flèches aux extrémités :

```
\draw[blue,->] (0,0) -- (1,1);
\draw[red,<->] (1,0) -- (2,1);
```

Surface remplie

Pour avoir une surface remplie (peinte), on utilise la commande

```
\fill chemin
```

Cette commande accepte la couleur comme paramètre. par exemple, pour tracer un disque gris :

```
\fill[gray] (0,0) circle (1cm)
```

On peut remplir avec un dégradé en utilisant la commande `\shade`. On peut passer des couleurs en argument :

- dégradé horizontal : `\shade[left color=couleur1, right color=couleur2];`
- dégradé vertical : `\shade[top color=couleur1, bottom color=couleur2];`
- dégradé vers l'extérieur : `\shade[inner color=couleur1, outer color=couleur2];`
- effet de balle : `\shade[ball color=couleur]`.

Utilisation des nœuds

Un nœud est une boîte qui peut contenir du texte. Pour obtenir un nœud aux coordonnées *x* et *y*, on utilise la commande suivante :

```
\node at (x,y) {du texte};
```

La boîte du nœud peut être remplie, son bord peut être tracé et on peut choisir la couleur du texte :

```
\node[fill=yellow,draw=black,text=blue] at (0,0) {du texte};
```

On peut aussi nommer un nœud pour y faire appel ensuite :

```
\node (premier) at (0,0) {du texte};
\node (second) at (2,2) {autre texte};
\draw[->] (premier) -- (second);
```

TikZ et Babel french/français

Utiliser TikZ avec la bibliothèque babel en français pose des problèmes car cette bibliothèque rend actifs les caractères { : ; ! ? }.

Pour pallier cette incompatibilité il suffit de rajouter au début d'une séquence TikZ :

```
\shorthandoff{:;!?};
```

Comme par exemple ici :

```
\begin{tikzpicture}
    \shorthandoff{:;!?};
    \draw (0,-2) arc (-90:-270:1);
\end{tikzpicture}
```

Si vous voulez faire ceci globalement pour tout les environnements {tikzpicture} il suffit de mettre au début du document :

```
\tikzset{every picture/.style={execute at begin picture={
   \shorthandoff{:;!?};}
}}
```

Voir aussi

Liens externes

- PGF (http://www.ctan.org/tex-archive/graphics/pgf/) [archive] sur la CTAN
- TikZ pour l'impatient (http://math.et.info.free.fr/TikZ/index.html) [archive] un manuel en français pour débuter rapidement
- TeXample.net (http://www.texample.net/) [archive] un site proposant de nombreux exemples de dessins et graphiques réalisés via TikZ

Dessiner avec LaTeX/Dessiner des molécules

Vous êtes invité à consulter d'abord la section *[[../../Écrire des formules chimiques|Écrire des formules chimiques]]*.

Nous présentons quelques solutions ici. La plus ancienne est ChemTeX; la plus performante est PPCHTeX; la plus récente qui combine souplesse, facilité d'utilisation et puissance est ChemFig.

Avec `chemfig`

Pour utiliser cette extension, il faut commencer par placer dans le préambule le code suivant :

```
\usepackage{chemfig}
```

On décrit le dessin selon une syntaxe simple, souple et intuitive. Le dessin est fait avec l'extension tikz qui se charge automatiquement. ChemFig fonctionne donc en mode dvi ou pdf.

La commande principale permettant de dessiner les molécules est

```
\chemfig{<code>}
```

L'argument *code* est la suite de caractères décrivant le dessin de la molécule selon les règles qui sont exposées dans le manuel de chemfig. Tout a été fait pour qu'il soit possible de dessiner le plus grand nombre de configurations de molécules chimiques, tout en privilégiant une syntaxe simple, souple et intuitive. Malgré tout, le *code* qui décrit le dessin en 2D de la molécule voit sa complexité augmenter proportionnellement à celle de la molécule à dessiner.

Avec `ppchtex`

Rappel

L'utilisation de l'extension nécessite la déclaration, dans le préambule, de `\usepackage{m-pictex,m-ch-en}`, éventuellement précédé de `\usepackage{etex}`.

L'extension PPCHTeX extension permet également de dessiner des molécules développées en deux dimensions, et de représenter les orientations des liaisons (liaisons tétraédriques du carbone, projection de Newman, voir aussi *Représentation des molécules*).

La forme générale est :

```
\startchemical
   \chemical[liaisons][groupements]
\stopchemical
```

La liste *liaisons* commence par la description du groupement au point de création, par exemple `ONE` pour des liaisons en étoile autour d'un groupement, et `SIX` pour un cycle hexagonal. Suivent ensuite les liaisons à tracer.

On peut ajouter des options :

```
\startchemical[options]
   \chemical[liaisons][groupements]
\stopchemical
```

ou encore

```
\setupchemical[option]
\startchemical
   \chemical[liaisons][groupements]
\stopchemical
```

Les options possibles sont :

- `frame=on` : trace un cadre autour du dessin ;
- `axis=on` : trace des axes gradués se coupant au centre ;
- `scale=small` : dessin plus petit ;
- `alternative=2` : traits plus fins ;

- `size=small` ou `big` : taille des caractères ;
- `character=\bf` : caractères gras ;
- `option=test` : trace un cadre autour du texte, pour vérifier l'alignement.

Structure `ONE`

Dans le cas de `ONE`, il s'agit des liaisons partant du groupement central ; on peut en mettre 8, la n°1 étant à droite, la numérotation se faisant dans le sens anti-trigonométrique.

```
6 7 8
 \|/
5-0-1
 /|\
4 3 2
```

On utilise la lettre `SB` pour une liaison simple, `DB` pour une double et `TB` pour une triple. Par exemple

```
ONE,SB1,SB3,SB5,SB7
```

va placer quatre liaisons simples en croix +. On pourrait également écrire

```
ONE,SB1357
```

Suit la liste des *positions* des groupements ou atomes, introduite par `Z`, la position 0 étant le centre.

Puis, le second crochet contient la liste des groupements en eux-mêmes, dans le même ordre, écrits avec la syntaxe chimique classique.

Par exemple, pour la formule développée du méthane :

```
\startchemical
   \chemical[ONE,SB1357,Z01357][C,H,H,H,H]
\stopchemical
```

- `SB1357` indique qu'il y a quatre liaisons situées en 1, 3, 5 et 7 ;
- `Z01357` indique qu'il y a cinq groupements situées en 0 (centre), 1, 3, 5 et 7 ;
- `[C,H,H,H,H]` est la liste des groupements : un atome de carbone à la position 0, et un atome d'hydrogène aux positions suivantes.

```
     (7)
      H
      |
(5) H-C-H (1)
      |
      H
     (3)
```

On peut aussi faire des liaisons en pointillé avec `SD` (D comme *dottted*), et des liaisons en trait épais avec `BB` *(bold bond)*. Par exemple, pour la projection de Newmann du méthane :

```
\startchemical
   \chemical[ONE,SD1,SB4,BB2,SB7,Z01247][C,H,H,H,H]
\stopchemical
     (7)
      H
      |
      C···H (1)
     / \
    H   H
   (4) (2)
```

Structure `SIX`

Si l'on utilise `SIX`, cela crée un hexagone. Les positions et les liaisons sont numérotées dans le sens horaire, la position 1 est en haut à droite, la liaison 1 est la liaison verticale à droite.

```
       6
 (L5) / \ (L6)
     5   1
(L4) |   | (L1)
     4   2
 (L3) \ / (L2)
```

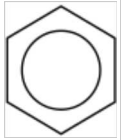
```
      3
```

On a deux notations :

- la notation explicite, pour laquelle on indique les atomes de carbone du cyles : les liaisons `SB`, `DB` et `TB` désignent les liaisons respectivement simple, double et triple entre les atomes de l'hexagone
- la notation simplifiée, pour laquelle on n'indique pas ces atomes (les traits sont plus longs et se rejoignent) : on utilise `B` *(bond)*, et on ajoute `EB` *(extra bond)* si l'on veut une liaison double.

La lettre `C` désigne la liaison délocalisée *(circle)*. On met le ou les numéros des liaisons concernés après la nature de la liaison, et si l'on n'utilise pas de numéro, cela désigne toutes les liaisons. Par exemple, pour le benzène en notation simplifiée

```
\startchemical
   \chemical[SIX,B,C]
\stopchemical
```


liaison délocalisée

le `B` tout seul indique que toutes les liaisons internes sont simples. Pour la forme de Kékulé :

```
\startchemical
   \chemical[SIX,B,EB246]
\stopchemical
```


forme de Kékulé

les liaisons 1, 3 et 5 sont simples, des 2, 4 et 6 sont doubles. On aurait aussi pû mettre `B1,B2,EB2,B3,B4,EB4,B5,B6,EB6`.

Pour mettre des atomes dans l'hexagone, on utilise `Z` ; il faut alors utiliser `SB` et `DB` (et non pas `B` et `EB`) :

```
\startchemical
   \chemical[SIX,SB,C,Z][C,C,C,C,C,C]
\stopchemical
```

```
\startchemical
   \chemical[SIX,SB135,DB246,Z][C,C,C,C,C,C]
\stopchemical
```

`Z0` désigne le centre, par exemple si l'on veut mettre une charge partielle positive `\oplus` ou `\delta +`.

Pour mettre des liaisons partant des atomes du cycle et allant vers des groupements extérieurs, on utilise `R` *(radical)* pour une liaison simple et `ER` *(extra radical)* pour une liaison double ; pour la projection de Newmann, on peut utiliser `RD` pour une liaison en pointillés *(radical dashed)* et `RB` pour une liaison en trait gras *(radical bold)*. Là encore, on met le ou les numéros des sites concernés après, et si l'on n'utilise pas de numéro, cela désigne toutes les liaisons. Si l'on a des atomes placés dans l'hexagone, il faut utiliser `SR` *(single radical)* ou `DR` *(doube radical)* pour que le trait ne se superpose pas au symbole.

Si l'on veut placer des groupements au bout des liaisons extérieures, on utilise `RZ`. Le ou les groupements sont indiqués dans le crochet après, dans l'ordre des `RZ`.

Par exemple, pour la structure explicite du benzène :

```
\startchemical
   \chemical[SIX,SB,C,Z,SR,RZ][C,C,C,C,C,C,H,H,H,H,H,H]
\stopchemical
```

pour la forme simplifiée du phénol

```
\startchemical
   \chemical[SIX,B,R6,RZ6][OH]
\stopchemical
```

L'option `FRONT` permet de tourner d'un sixième de tour.

```
      (4)
     4___5
(3) /     \ (5)
   3       6
(2) \ ___ / (6)
     2   1
      (1)
```

Cette forme ne peut pas être utilisée pour les cycles aromatiques (la liaison `C` ne fonctionne pas), mais pour le cyclohexane.

Alors, `+R` et `-R` désignent des liaisons respectivement vers le haut et vers le bas, `+RZ` et `-RZ` désignent des groupements respectivement vers le haut et vers le bas. On peut séparer les parties de la molécules en plusieurs commandes `\chemical` pour simplifier, par exemple

```
\startchemical
   \chemical[SIX,FRONT,B] % structure de base
   \chemical[SIX,FRONT,+R,+RZ][liste de groupements] % groupements en haut
   \chemical[SIX,FRONT,-R,-RZ][liste de groupements] % groupements en bas
\stopchemical
```

On peut utiliser des liaisons raccourcies vers la gauche, `-SB`, ou vers la droite, `+SB`. En raccourcissant une liaison vers la gauche et une autre vers la droite, cela laisse la place de mettre un atome dans l'hexagone, par exemple

```
\startchemical
   \chemical[SIX,FRONT,B1236,+SB4,-SB5,Z5][O]
\stopchemical
```

laisse de la place en position 5 pour placer un atome (ici d'oxygène).

```
  __O
 /   \
 \___/
```

Autres structures

Les autres structures sont :

- `THREE` : triangle ;
- `FOUR` : carré ;
- `FIVE` : pentagone ;
- `FIVE,FRONT` : pentagone avec des liaisons vers le haut et le bas ;
- `EIGHT` : octogone ;
- `CARBON` : carbone tétraédrique (hybridation sp^3) en perspective ;
- `NEWMAN` : représentation de Newman ;
- `CHAIR` : cyclohexane en conformation chaise.

Mise en forme des groupements

On peut faire varier l'alignement du texte des groupements, par exemple pour éviter la collision :

- `\SR{`*groupement*`}` pour aligner le texte à droite par rapport au point d'implantation (le texte se trouve donc à gauche de ce point) ;
- `\SL{`*groupement*`}` pour aligner le texte à gauche ;
- `\SC{`*groupement*`}` pour centrer le texte.

On peut rajouter du texte aux groupements, avec la syntaxe

```
\T{texte}{groupement}
```

On peut indiquer la charge électrique avec

```
\-{nombre}{groupement} % charge négative
\+{nombre}{groupement} % charge positive
```

par exemple

```
\startchemical
   \chemical[ONE,SB1257,Z01357][\T{1}{C},H,H,H,H]
\stopchemical
```

ajoute un « 1 » au dessus du carbone.

Chaîne de structures

On peut enchaîner les structures en créant une structure dans une autre. La sous-structure est définie dans le premier crochet, avec les position des atomes ; elle commence par `PB:`*`position`* *(picture begin)* et se termine par `PE` *(picture end)*. Par exemple, pour représenter une molécule de dioxygène :

```
\startchemical
```

```
   \chemical[ONE,DB1,Z0,PB:Z1,ONE,Z0,PE][O,O]
\stopchemical
```

```
O=O
```

Notez que l'on aurait pu simplement faire `\chemical[ONE,DB1,Z01][O,O]` ou bien `\chemical{0,--,0}` (sans `\start/stopchemical` pour cette dernière solution), c'est juste un exemple simplifié. Cependant, cette solution est utile si l'on veut placer les doublets non-liants (voir ci-après).

Par exemple, pour le phénol :

```
\startchemical
   \chemical[SIX,B,C,R6,PB:RZ6,ONE,SB7,Z07,PE][O,H]
\stopchemical
```

```
  H
  |
  O
  |
 / \
|   |
 \ /
```

Comme précédemment, il peut être intéressant de séparer la description en plusieurs parties. Par exemple, pour l'éthène :

```
\startchemical
   \chemical[ONE,DB1,SB46,Z046][C,H,H]
   \chemical[ONE,PB:Z1,ONE,SB28,Z028,PE][C,H,H]
\stopchemical
```

```
H       H
 \     /
  C = C
 /     \
H       H
```

Si les deux sous-structures sont de même type (`ONE` ou `SIX`), on peut simplement utiliser `MOV` avec la direction pour indiquer que l'on décale la structure. Par exemple, toujours pour l'éthène :

```
\startchemical
   \chemical[ONE,DB1,SB46,Z046][C,H,H]
   \chemical[ONE,MOV1,SB28,Z028,PE][C,H,H]
\stopchemical
```

Polymères

On dispose d'une liaison `OE` *(open ended)*, qui indique que la structure continue au-delà de la liaison.

Représentation de Lewis

On peut utiliser des liaisons spéciales pour la représentation de Lewis :

- électron célibataire : `ES` *(extra single)* ;
- deux électrons célibataires : `ED` *(extra double)* ;
- deux électrons célibataires : `ET` *(extra triple)* ;
- doublet non liant : `EP` *(extra pair)*.

Par exemple, pour le dioxygène

```
\startchemical
   \chemical[ONE,DB1,EP46,Z0,PB:Z1,ONE,EP82,Z0,PE][O,O]
\stopchemical
/     \
 O = O
\     /
```

Documentation

On pourra se reporter à la documentation de l'extension, par exemple

- *PPCHTEX, a macropackage for typesetting chemical structure formulas* (http://www.pragma-ade.com/general/manuals/mp-ch-en.pdf) [archive]

du site officiel Pragma ADE (http://www.pragma-ade.com/overview.htm) [archive]
- le document PDF (http://www.mcs.vuw.ac.nz/technical/software/latexdoc/context/ppchtex/mp-ch-en.pdf) [archive] (756 ko) à la page teTeX Documentation Guide (http://www.mcs.vuw.ac.nz/technical/software/latexdoc/) [archive],

ainsi qu'à la description succincte dans Bitouzé et Charpentier[1] p. 210.

Avec le mode mathématiques

Outre ce qui est écrit dans *[[../../Écrire des formules chimiques#Avec le mode mathématiques|Écrire des formules chimiques > Avec le mode mathématiques]]*, ajoutons que :

- les liaisons verticales peuvent être obtenues avec `|` (ou `\vert`) pour les liaisons simples, et `\|` (ou `\Vert`) pour les doubles ;
- pour placer des éléments les uns par rapport aux autres, on peut placer le tout dans un environnement `picture` et placer les éléments avec des instructions `\put`.

Avec `chemtex`

Les molécules se définissent simplement en mode mathématique, et éventuellement dans un environnement `picture`, comme évoqué ci-dessus.

Pour dessiner une branche de molécule en Y tourné vers la droite, on utilise :

```
\cright{G1}{L1}{G0}{L2}{G2}{L3}{G3}
```

```
      G2
   L1  / L2
G1 - G0
      \ L3
      G3
```

Les différentes parties sont ensuite placées dans un environnement `picture` avec des instructions `\put`. ChemTeX est donc une sorte d'aide au dessin avec LaTeX.

Notes

1. D. Bitouzé et J.-C. Charpentier, *LaTeX, synthèse et cours*, éd. Pearson Education, 2006, ISBN 2-7440-7187-0

Voir aussi

Liens externes

Le package chemfig (http://tug.ctan.org/tex-archive/macros/latex/contrib/chemfig/) [archive] semble également très prometteur pour le dessin de molécules chimiques en 2D, autant du point de vue de la syntaxe que des possibilités de dessin qu'il offre. Consulter la documentation (http://tug.ctan.org/tex-archive/macros/latex/contrib/chemfig/chemfig_doc_fr.pdf) [archive] en français (pdf, 810kio).

Dessiner avec LaTeX/Réalisation de graphiques mathématiques

Comme indiqué précédemment, LaTeX offre des possibilités, limitées, d'intégrer des graphiques à partir de descriptions textuelles. Par ailleurs, un certain nombre d'extensions de LaTeX ont été créées afin de combler ces faiblesses. Dans ce chapitre, vous apprendrez à utiliser ces possibilités pour réaliser des graphiques mathématiques.

Vue d'ensemble

L'environnement `picture`, amélioré par les extensions `pict2e` et `graphicx`, permet de tracer des segments, des rectangles, des cercles, et des courbes de bézier quadratiques et cubiques. Les ellipses les chaînettes peuvent être approchées d'une manière satisfaisante par des courbes quadratiques de Bézier, bien que ceci puisse exiger un travail mathématique assez conséquent. Si, en outre, un langage de programmation comme Java est employé pour produire des fichiers LaTeX de blocs `\qbezier`, l'environnement d'image *picture* devient tout à fait puissant.

Bien que la programmation d'images directement en LaTeX soit sévèrement restreinte, et souvent fatiguante, il y a tout de même des raisons de procéder ainsi. Les documents ainsi produits sont de petite taille en ce qui concerne le nombre d'octets, et il n'y a aucun fichier graphique supplémentaire à adjoindre au fichier Latex.

Des extensions comme `epic`, `eepic` ou `pstricks` aident à effacer les limitations entravant l'environnement original *picture*, et renforcent considérablement la puissance graphique de Latex.

Tandis que les deux anciennes extensions agrémentent simplement l'environnement d'image, l'extension `pstricks` possède son propre environnement de dessin `pspicture`. La puissance de `pstricks` provient du fait que cette extension profite des possibilités étendues du langage PostScript. En outre, de nombreuses extensions ont été écrites dans des buts spécifiques. L'une de ces extensions *XY-pic*, sera décrite à la fin de ce chapitre. Une grande variété d'extensions sont décrites en détail dans le livre *The LaTeX Graphics Companion* (qu'il ne faut pas confondre avec *The LaTeX Companion*).

Sans doute, l'outil graphique le plus puissant du système LaTeX est MetaPost, le jumeau de METAFONT de Donald E. Knuth. MetaPost est doté du langage de programmation très puissant et mathématiquement sophistiqué de METAFONT. Contrairement à METAFONT, qui génère des fichiers au format bitmap, MetaPost produit des fichiers PostScript encapsulés, qui peuvent être importés dans un fichier Latex. Pour une introduction à cet outil, vous pouvez consulter le manuel *A User's Manual for MetaPost*. Une discussion très complète des stratégies Latex et TEX pour les graphiques (et les polices) peut être trouvée dans *TEX Unbound*.

L'environnement *picture*

Les commandes de base

Un environnement `picture` est disponible dans toute distribution LaTeX, sans avoir besoin d'inclure une extension externe. La création d'une image au moyen de cet environnement s'effectue avec l'une de ces deux commandes:

```
\begin{picture}(x, y) ... \end{picture}
```

ou

```
\begin{picture}(x, y)(x0, y0) ... \end{picture}
```

Les nombres *x*, *y*, *x0*, *y0* se rapportent à `\unitlength`, qui peut être modifiée à tout moment (mais pas dans l'environnement picture) avec une commande telle que

```
\setlength{\unitlength}{1.2cm}
```

La valeur par défaut de `\unitlength` est de `1pt`. Le premier couple, (x, y), représente la taille de l'espace rectangulaire réservé à l'image dans le document. Le second couple, (x_0, y_0), correspond aux coordonnées arbitraires du coin en bas à gauche du rectangle renfermant l'image.

Beaucoup de commandes de dessin ont l'une ou l'autre de ces formes:

```
\put(x, y){objet}
```

ou

```
\multiput(x, y)(dx, dy){n}{objet}
```

La commande pour dessiner une courbe de Bézier est une exception. La commande est de la forme:

```
\qbezier(x1, y1)(x2, y2)(x3, y3)
```

Segments de droite

Les segments de droite peuvent être dessinés avec la commande:

```
\put(x, y){\line(x1, y1){longueur}}
```

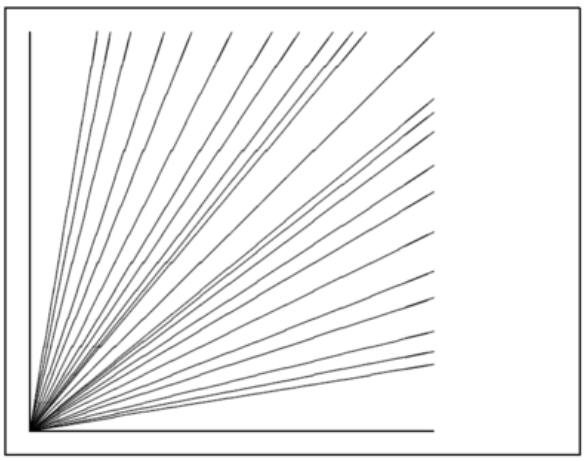
La commande `\line` possède deux paramètres:

1. un vecteur de direction,
2. une longueur.

De manière native, les composantes du vecteur de direction sont restreintes aux entiers de l'ensemble $\{-6, -5, \ldots, 5, 6\}$ et doivent être premières entre elles (sans diviseur commun à l'exception de 1). Cette restriction a été levée par `pict2e`.

La figure suivante illustre toutes les 25 valeurs de pente possibles (sans `pict2e`) dans le premier quadrant. La longueur du segement est relative à l'unité de longueur `\unitlength`. Le paramètre longueur représente l'ordonnée dans le cas d'un segment « vertical », et l'abscisse dans tous les autres cas et ne représente donc pas la norme du vecteur de direction, ni la longueur du segment.

```
\setlength{\unitlength}{5cm}
\begin{picture}(1,1)
\put(0,0){\line(0,1){1}}
\put(0,0){\line(1,0){1}}
\put(0,0){\line(1,1){1}}
\put(0,0){\line(1,2){.5}}
\put(0,0){\line(1,3){.3333}}
\put(0,0){\line(1,4){.25}}
\put(0,0){\line(1,5){.2}}
\put(0,0){\line(1,6){.1667}}
\put(0,0){\line(2,1){1}}
\put(0,0){\line(2,3){.6667}}
\put(0,0){\line(2,5){.4}}
\put(0,0){\line(3,1){1}}
\put(0,0){\line(3,2){1}}
\put(0,0){\line(3,4){.75}}
\put(0,0){\line(3,5){.6}}
\put(0,0){\line(4,1){1}}
\put(0,0){\line(4,3){1}}
\put(0,0){\line(4,5){.8}}
\put(0,0){\line(5,1){1}}
\put(0,0){\line(5,2){1}}
\put(0,0){\line(5,3){1}}
\put(0,0){\line(5,4){1}}
\put(0,0){\line(5,6){.8333}}
\put(0,0){\line(6,1){1}}
\put(0,0){\line(6,5){1}}
\end{picture}
```



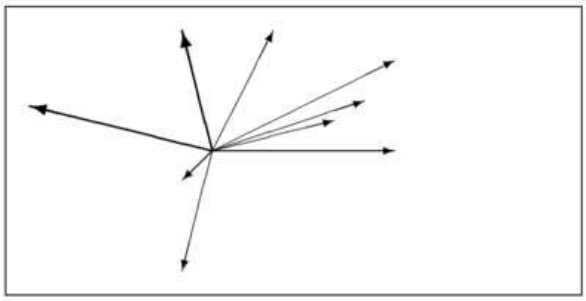
Flèches

Les flèches se dessinent avec la commande:

```
\put(x, y){\vector(x1, y1){longueur}}
```

De manière native, les composantes du vecteur de direction pour les flèches subissent encore plus de restrictions que celles des vecteurs de segment de droite puisqu'elles ne peuvent prendre que des valeurs entières de l'ensemble $\{-4, -3, \ldots, 3, 4\}$. Cette restriction est également levée par `pict2e` Remarquez l'effet de la commande `\thicklines` sur les deux flèches pointant le haut et la gauche.

```
\setlength{\unitlength}{0.75mm}
\begin{picture}(60,40)
\put(30,20){\vector(1,0){30}}
\put(30,20){\vector(4,1){20}}
\put(30,20){\vector(3,1){25}}
\put(30,20){\vector(2,1){30}}
\put(30,20){\vector(1,2){10}}
\thicklines
\put(30,20){\vector(-4,1){30}}
\put(30,20){\vector(-1,4){5}}
\thinlines
\put(30,20){\vector(-1,-1){5}}
\put(30,20){\vector(-1,-4){5}}
\end{picture}
```



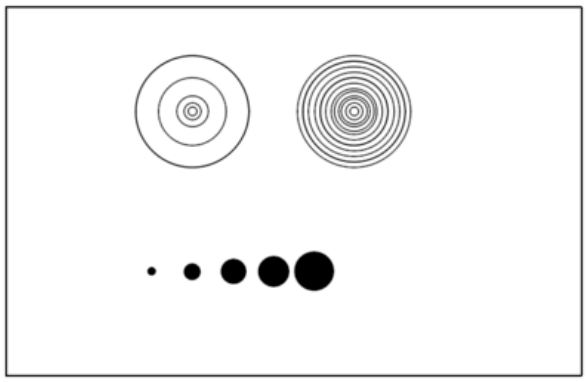
Cercles

La commande

```
\put(x, y){\circle{diamètre}}
```

dessine un cercle de centre (x, y) et de diamètre (pas le rayon) *diamètre*. L'environnement *picture* n'autorise que des diamètres d'au plus 14 mm environ, et même en dessous de cette limite, tous les diamètres ne sont pas possibles. La commande `\circle*` réalise des disques (cercles pleins). Comme dans le cas des segments de droite, il est possible de recourir à des extensions supplémentaires, telles que `pict2e`, `eepic` ou `pstricks`.

```
\setlength{\unitlength}{1mm}
\begin{picture}(60, 40)
\put(20,30){\circle{1}}
\put(20,30){\circle{2}}
\put(20,30){\circle{4}}
\put(20,30){\circle{8}}
\put(20,30){\circle{16}}
\put(20,30){\circle{32}}
\put(40,30){\circle{1}}
\put(40,30){\circle{2}}
\put(40,30){\circle{3}}
\put(40,30){\circle{4}}
\put(40,30){\circle{5}}
\put(40,30){\circle{6}}
\put(40,30){\circle{7}}
\put(40,30){\circle{8}}
\put(40,30){\circle{9}}
\put(40,30){\circle{10}}
\put(40,30){\circle{11}}
\put(40,30){\circle{12}}
\put(40,30){\circle{13}}
\put(40,30){\circle{14}}
\put(15,10){\circle*{1}}
\put(20,10){\circle*{2}}
\put(25,10){\circle*{3}}
\put(30,10){\circle*{4}}
\put(35,10){\circle*{5}}
\end{picture}
```

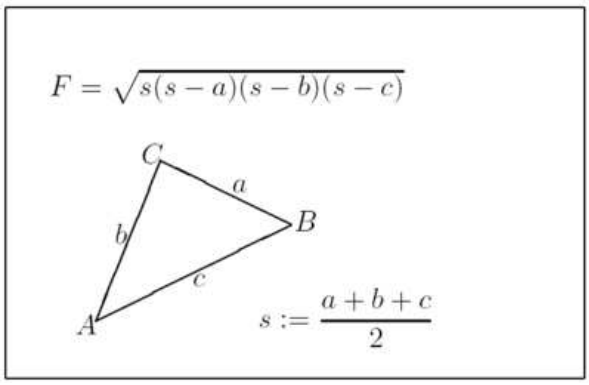


Il y a aussi une possibilité intéressante dans l'environnement *picture*. Si on n'a pas peur d'effectuer les calculs nécessaires (ou de les faire au moyen d'un programme), des cercles et des ellipses arbitrairement petits peuvent être raccordés ensemble à partir de courbes quadratiques de Bézier. Voir *Graphics in LaTeX2e* pour des exemples et les fichiers sources en Java.

Formules et texte

Comme cet exemple le montre, du texte et des formules peuvent être écrits dans l'environnement *picture* avec la commande `\put` de façon habituelle:

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,5)
\thicklines
\put(1,0.5){\line(2,1){3}}
\put(4,2){\line(-2,1){2}}
\put(2,3){\line(-2,-5){1}}
\put(0.7,0.3){$A$}
\put(4.05,1.9){$B$}
\put(1.7,2.95){$C$}
\put(3.1,2.5){$a$}
\put(1.3,1.7){$b$}
\put(2.5,1.05){$c$}
\put(0.3,4){$F=
\sqrt{s(s-a)(s-b)(s-c)}$}
\put(3.5,0.4){$\displaystyle
s:=\frac{a+b+c}{2}$}
\end{picture}
```



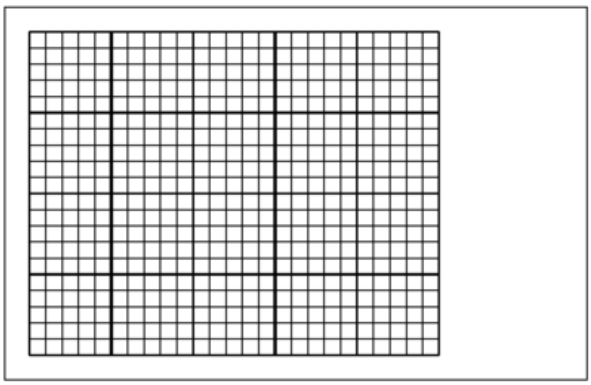
`\multiput` et `\linethickness`

La commande

```
\multiput(x, y)(dx, dy ){n}{objet}
```

possède quatre paramètres: le point de départ, le vecteur de translation d'un objet vers le suivant, le nombre d'objets, et l'objet à dessiner. La commande `\linethickness` modifie les segments de droite horizontaux et verticaux, mais jamais les segments obliques, ni les cercles. Cependant, elle s'applique aux courbes de Bézier quadratiques!

```
\setlength{\unitlength}{2mm}
\begin{picture}(30,20)
\linethickness{0.075mm}
\multiput(0,0)(1,0){26}%
{\line(0,1){20}}
\multiput(0,0)(0,1){21}%
{\line(1,0){25}}
\linethickness{0.15mm}
\multiput(0,0)(5,0){6}%
{\line(0,1){20}}
\multiput(0,0)(0,5){5}%
{\line(1,0){25}}
\linethickness{0.3mm}
\multiput(5,0)(10,0){2}%
{\line(0,1){20}}
\multiput(0,5)(0,10){2}%
{\line(1,0){25}}
\end{picture}
```



Ovales

La commande

```
\put(x, y){\oval(w, h)}
```

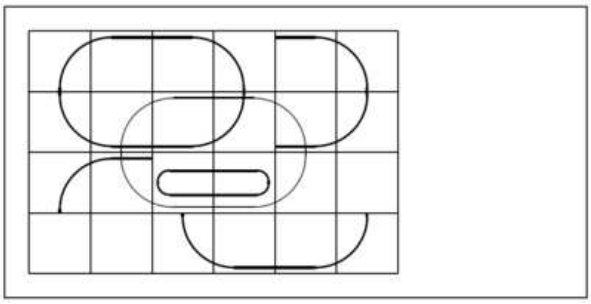
ou

```
\put(x, y){\oval(l, h)[position]}
```

dessine un ovale centré en le point de coordonnées (x, y) et ayant une largeur w et une hauteur h.

Les paramètres optionnels de position b, t, l, r se rapportent respectivement au bas, au haut, à la gauche et à la droite et peuvent être combinés, comme l'exemple le montre. L'épaisseur des lignes peut être modifiée par deux type de commandes: `\linethickness{`*`longueur`*`}` et `\thinlines` d'une part et `\thicklines` d'autre part. Tandis que `\linethickness{`*`longueur`*`}` ne s'applique qu'à des lignes verticales et horizontales (et les courbes de Bézier quadratiques), `\thinlines` et `\thicklines` s'appliquent à des segments de droite obliques, aux cercles et aux ovales.

```
\setlength{\unitlength}{0.75cm}
\begin{picture}(6,4)
\linethickness{0.075mm}
\multiput(0,0)(1,0){7}%
{\line(0,1){4}}
\multiput(0,0)(0,1){5}%
{\line(1,0){6}}
\thicklines
\put(2,3){\oval(3,1.8)}
\thinlines
\put(3,2){\oval(3,1.8)}
\thicklines
\put(2,1){\oval(3,1.8)[tl]}
\put(4,1){\oval(3,1.8)[b]}
\put(4,3){\oval(3,1.8)[r]}
\put(3,1.5){\oval(1.8,0.4)}
\end{picture}
```



Usages multiples de boîtes d'image

Une image peut être « déclarée » grâce à la commande:

```
\newsavebox{nom}
```

puis « définie » par

```
\savebox{nom}(largeur, hauteur)[position]{contenu}
```

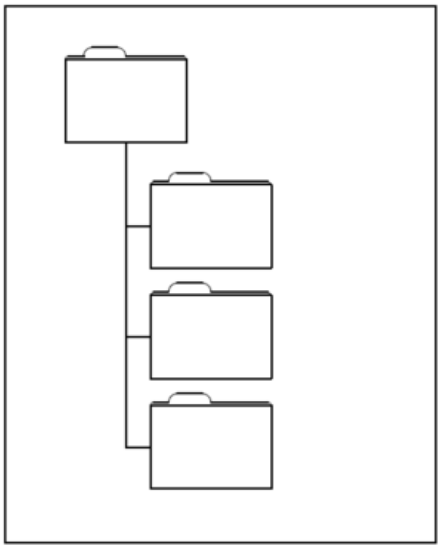
et finalement dessinée le plus souvent par la commande

```
\put(x, y){\usebox{nom}}
```

Le paramètre optionnel de position a pour utilité de définir la « point d'ancrage » de la boîte de sauvegarde. Dans l'exemple, il est fixé à « bl » ce qui a pour conséquence de placer le point d'ancrage dans le coin en bas et à gauche de la boîte. Les autres indicateurs de position représentent le haut et la droite.

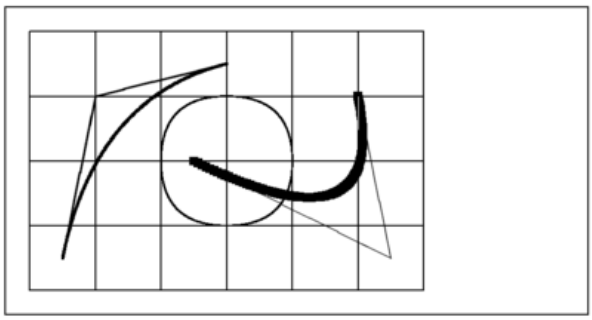
Le paramètre *nom* se rapporte à une zone de stockage binaire de Latex et est donc une commande (ce qui explique la barre oblique inversée dans l'exemple). Des images rectangulaires peuvent y être placées : Dans cet exemple, `\foldera` est employé dans la définition de `\folderb`. La commande `\oval` a dû être employée étant donné que la commande `\line` ne fonctionne pas lorsque la longueur du segment est inférieur à environ 3 millimètres.

```
\setlength{\unitlength}{0.5mm}
\begin{picture}(120,168)
\newsavebox{\foldera}
\savebox{\foldera}
  (40,32)[bl]{% definition
  \multiput(0,0)(0,28){2}
    {\line(1,0){40}}
  \multiput(0,0)(40,0){2}
    {\line(0,1){28}}
  \put(1,28){\oval(2,2)[tl]}
  \put(1,29){\line(1,0){5}}
  \put(9,29){\oval(6,6)[tl]}
  \put(9,32){\line(1,0){8}}
  \put(17,29){\oval(6,6)[tr]}
  \put(20,29){\line(1,0){19}}
  \put(39,28){\oval(2,2)[tr]}
}

\newsavebox{\folderb}
\savebox{\folderb}
  (40,32)[l]{% definition
  \put(0,14){\line(1,0){8}}
  \put(8,0){\usebox{\foldera}}
}

\put(34,26){\line(0,1){102}}
\put(14,128){\usebox{\foldera}}
\multiput(34,86)(0,-37){3}
{\usebox{\folderb}}
\end{picture}
```



Courbes de Bézier quadratiques

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,4)
\linethickness{0.075mm}
\multiput(0,0)(1,0){7}
{\line(0,1){4}}
\multiput(0,0)(0,1){5}
{\line(1,0){6}}
\thicklines
\put(0.5,0.5){\line(1,5){0.5}}
\put(1,3){\line(4,1){2}}
\qbezier(0.5,0.5)(1,3)(3,3.5)
\thinlines
\put(2.5,2){\line(2,-1){3}}
\put(5.5,0.5){\line(-1,5){0.5}}
\linethickness{1mm}
\qbezier(2.5,2)(5.5,0.5)(5,3)
\thinlines
\qbezier(4,2)(4,3)(3,3)
\qbezier(3,3)(2,3)(2,2)
\qbezier(2,2)(2,1)(3,1)
\qbezier(3,1)(4,1)(4,2)
\end{picture}
```



Comme cet exemple le montre, découper un cercle en 4 courbes de Bézier quadratiques n'est pas satisfaisant. Au moins huit découpages sont nécessaires. La figure montre aussi l'effet de la commande `\linethickness` sur les lignes verticales et horizontales, et l'effet des commandes `\thinlines` et `\thicklines` sur les segments de droite obliques. On peut remarquer de plus que ces deux sortes de commandes affectent les courbes de Bézier, chaque commande venant annuler l'effet des précédentes.

Soient $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$ les points terminaux et m_1, m_2 les pentes respectives, de courbes de Bézier quadratiques. Le point de contrôle intermédiaire $S = (x, y)$ est donné par les relations:

$$\begin{cases} x = \frac{m_2 x_2 - m_1 x_1 - (y_2 - y_1)}{m_2 - m_1} \\ y = y_i + m_i(x - x_i); \quad (i = 1, 2) \end{cases}$$

Voir *Graphics in LaTeX2e* pour un programme en Java qui génère les commandes `\qbezier` nécessaires à l'obtention de cercles.

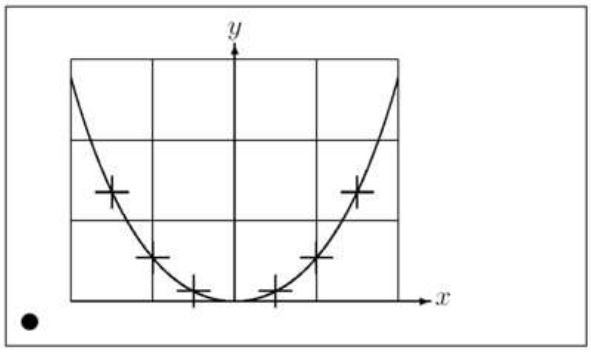
Chaînette

Dans cette figure, chaque moitié symétrique de la chaînette $y = \mathrm{ch}x - 1$ est approchée par une courbe de Bézier quadratique. La moitié de droite de la courbe se termine au point de coordonnées $(2; 2,7622)$, la pente en ce point ayant la valeur $m = 3,6269$. En utilisant à nouveau la relation (*), nous pouvons calculer les coordonnées des points de contrôles intermédiaires. Elles s'avèrent être $(1,2384; 0)$ et $(-1,2384; 0)$. Les croix indiquent les points de la chaînette réelle. L'erreur est à peine visible, étant inférieur à un pour cent. Cet exemple illustre l'utilisation du paramètre facultatif de la commande `\begin{picture}`. L'image est définie dans un système de coordonnées mathématiques adéquat, tandis que par la commande

```
\begin{picture}(4.3,3.6)(-2.5,-0.25)
```

son coin inférieur gauche (marqué par le disque noir) a pour coordonnées $(-2,5; -0,25)$.

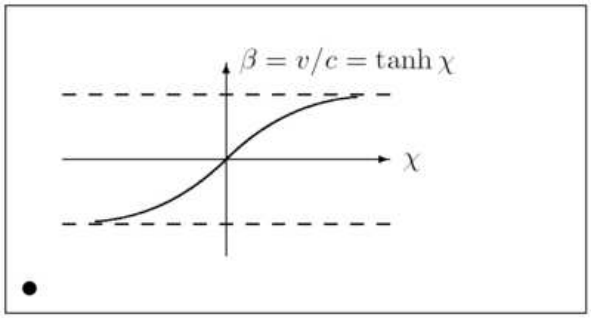
```
\setlength{\unitlength}{1cm}
\begin{picture}(4.3,3.6)(-2.5,-0.25)
\put(-2,0){\vector(1,0){4.4}}
\put(2.45,-.05){$x$}
\put(0,0){\vector(0,1){3.2}}
\put(0,3.35){\makebox(0,0){$y$}}
\qbezier(0.0,0.0)(1.2384,0.0)
(2.0,2.7622)
\qbezier(0.0,0.0)(-1.2384,0.0)
(-2.0,2.7622)
\linethickness{.075mm}
\multiput(-2,0)(1,0){5}
{\line(0,1){3}}
\multiput(-2,0)(0,1){4}
{\line(1,0){4}}
\linethickness{.2mm}
\put( .3,.12763){\line(1,0){.4}}
\put(.5,-.07237){\line(0,1){.4}}
\put(-.7,.12763){\line(1,0){.4}}
\put(-.5,-.07237){\line(0,1){.4}}
\put(.8,.54308){\line(1,0){.4}}
\put(1,.34308){\line(0,1){.4}}
```



```
\put(-1.2,.54308){\line(1,0){.4}}
\put(-1,.34308){\line(0,1){.4}}
\put(1.3,1.35241){\line(1,0){.4}}
\put(1.5,1.15241){\line(0,1){.4}}
\put(-1.7,1.35241){\line(1,0){.4}}
\put(-1.5,1.15241){\line(0,1){.4}}
\put(-2.5,-0.25){\circle*{0.2}}
\end{picture}
```

Vitesse dans la théorie de la relativité

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,4)(-3,-2)
\put(-2.5,0){\vector(1,0){5}}
\put(2.7,-0.1){$\chi$}
\put(0,-1.5){\vector(0,1){3}}
\multiput(-2.5,1)(0.4,0){13}
{\line(1,0){0.2}}
\multiput(-2.5,-1)(0.4,0){13}
{\line(1,0){0.2}}
\put(0.2,1.4)
{$\beta=v/c=\tanh\chi$}
\qbezier(0,0)(0.8853,0.8853)
(2,0.9640)
\qbezier(0,0)(-0.8853,-0.8853)
(-2,-0.9640)
\put(-3,-2){\circle*{0.2}}
\end{picture}
```



Les points de contrôle des deux courbes de Bézier ont été calculés avec les formules (*). La branche positive est déterminée par $P_1 = (0,0)$, $m_1 = 1$ et $P_2 = (2, \mathrm{th}2)$, $m_2 = 1/\mathrm{ch}^2 2$. À nouveau la figure est définie par rapport à un système de coordonnées adapté, et le coin inférieur gauche a pour coordonnées $(-3,-2)$ (disque noir).

Créer une extension ou une classe

Nous avons vu précédemment que l'on pouvait placer le préambule (appels d'extensions et commandes personnelles) dans un fichier `.tex` et appeler celui-ci avec la commande `\input`, placée entre la déclaration de classe et le `\begin{document}`.

Vous avez également la possibilité de créer votre propre classe ou votre propre extension.

Dans un premier temps, on peut voir une extension ou une classe comme un fichier de préambule, mais le potentiel est beaucoup plus important.

Extension personnelle

Une extension *(package)* est un fichier texte dont le nom se termine par `.sty`. Si son contenu est similaire à un fichier `.tex`, il existe quelques différences :

- il doit commencer par la déclaration du format de LaTeX pour lequel il a été conçu :
  `\NeedsTeXFormat{LaTeX2e}` pour LaTeX-2ε ;
- il faut indiquer que c'est une extension et indiquer le nom du fichier (sans le `.sty`) :
  `\ProvidesPackage{`*`NomDuFichier`*`}` ;
  il est recommandé d'indiquer des informations comme le numéro de version et la date afin de faciliter la mise à jour :
  `\ProvidesPackage{`*`NomDuFichier`*`}[`*`informations sur la version`*`]`.

Si l'on veut appeler une extension au sein de cette extension, on n'utilise pas `\usepackage` mais `\RequirePackage`.

Voici un exemple d'extension élémentaire reprenant simplement le préambule général des fichiers `.tex` :

Fichier `monextension.sty`

```
\NeedsTeXFormat{LaTeX2e}

\ProvidesPackage{monextension}[2007/09/19 Extension personnelle, V1.0]

% extensions

\RequirePackage[latin1]{inputenc}
\RequirePackage[T1]{fontenc}
\RequirePackage{lmodern}
\RequirePackage{graphicx}
\RequirePackage[frenchb]{babel}

% commandes personnelles

\newcommand{\langue}{\emph} % mots en langues étrangères
\newcommand{\citital}{\emph} % citation en italique
\newcommand{\nomprog}{\texttt} % nom de programme en police teletype
```

Le fichier `.tex` sera alors typiquement

```
\documentclass[a4paper, 11pt]{article}

\usepackage{monextension}

\begin{document}
...
```

Classe personnelle

Une classe *(class)* est un fichier texte dont le nom se termine par `.cls`. Si son contenu est similaire à un fichier `.tex`, il existe quelques différences :

- il doit commencer par la déclaration du format de LaTeX pour lequel il a été conçu :
  `\NeedsTeXFormat{LaTeX2e}` pour LaTeX-2ε ;
- il faut indiquer que c'est une classe et indiquer le nom du fichier (sans le `.cls`) :
  `\ProvidesClass{`*`NomDuFichier`*`}` ;
  il est recommandé d'indiquer des informations comme le numéro de version et la date afin de faciliter la mise à jour :
  `\ProvidesClass{`*`NomDuFichier`*`}[`*`informations sur la version`*`]`
  les informations sur la version doivent commencer par un chiffre, par exemple une date ou un numéro de version.

À moins de tout construire de zéro, on fait appel à une classe préexistante, à laquelle on va ajouter des éléments ou définir des paramètres. On utilise pour cela la commande `\LoadClass`, qui s'utilise comme `\documentclass` d'un fichier `.tex`.

Comme pour les extensions personnelles, si l'on veut appeler une extension au sein de cette classe, on n'utilise pas `\usepackage` mais `\RequirePackage`.

Voici un exemple de classe élémentaire reprenant simplement le préambule général des fichiers `.tex` :

Fichier `maclasse.cls`

```
\NeedsTeXFormat{LaTeX2e}

\ProvidesClass{maclasse}[2007/09/19 Classe personnelle, V1.0]

% classe de base

\LoadClass[a4paper, 11pt]{article}

% extensions

\RequirePackage[latin1]{inputenc}
\RequirePackage[T1]{fontenc}
\RequirePackage{lmodern}
\RequirePackage{graphicx}
\RequirePackage[frenchb]{babel}

% commandes personnelles

\newcommand{\langue}{\emph} % mots en langues étrangères
\newcommand{\citital}{\emph} % citation en italique
\newcommand{\nomprog}{\texttt} % nom de programme en police teletype
```

Le fichier `.tex` sera alors typiquement

```
\documentclass{maclasse}

\begin{document}
...
```

Gestion des options

On peut passer des options à une classe ou à une extension.

Options s'appliquant aux extensions appelées

La classe ou l'extension personnelle peut faire appel à d'autres extensions. Dans ce cas-là, les options peuvent être définies « en dur » comme nous l'avons fait ci-dessus, mais les options peuvent aussi être transmises lors de l'appel de l'extension ou de la classe personnelle.

Par exemple, si l'on veut laisser le choix de la langue, on utilise dans le fichier `.cls` ou `.sty`

```
\RequirePackageWithOptions{babel}
```

le fichier `.tex` contiendra alors

```
\usepackage[frenchb]{monextension}
```

ou bien

```
\documentclass[frenchb]{maclasse}
```

Options s'appliquant à la classe appelée

De même, lorsque l'on appelle une classe standard dans une classe personnelle, on peut laisser le choix des options de classe dans le fichier `.tex`, par exemple, pour le fichier `.cls` :

```
\LoadClassWithOptions{article}
```

et pour le fichier `.tex` :

```
\documentclass[a4paper, 11pt]{maclasse}
```

Options personnelles

Vous pouvez aussi définir vos propres options. Il suffit pour cela d'utiliser, dans le fichier `.sty` ou `.cls` :

```
\DeclareOption{NomDeLOption}{commandes}
\ProcessOptions
```

(ne pas oublier le `\ProcessOptions` à la fin de la déclaration des options, sinon on obtient des messages d'erreurs.)

Par exemple, dans le fichier `.sty` ou `.cls`

```
\RequirePackage{geometry}

\DeclareOption{petitemarge}{\geometry{lmargin=1cm,rmargin=1cm}}
\DeclareOption{grandemarge}{\geometry{lmargin=3cm,rmargin=3cm}}
```

et ensuite avoir dans le fichier `.tex`

```
\documentclass[petitemarge]{maclasse}
```

ou bien

```
\usepackage[petitemarge]{monextension}
```

Les commandes `\PassOptionsToPackage` et `\PassOptionsToclass` permettent de passer des options à une extension ou à une classe :

```
\PassOptionsToPackage{option}{extension}
\PassOptionsToClass{option}{classe}
```

par exemple

```
\RequirePackage{geometry}

\DeclareOption{petitemarge}{%
   \PassOptionsToPackage{lmargin=1cm,rmargin=1cm}{geometry}}
\DeclareOption{grandemarge}{%
   \PassOptionsToPackage{lmargin=3cm,rmargin=3cm}{geometry}}
```

On peut placer les commandes dans un fichier à part, portant l'extension `.clo`, et donc utiliser

```
\DeclareOption{petitemarge}{\input{petitemarge.clo}}
```

Voir aussi

Dans votre installation LaTeX

- le document `clsguide (http://mirror.ctan.org/macros/latex/doc/clsguide.pdf)` `[archive]` : rechercher ce nom de fichier sur le disque dur, ou bien taper `texdoc clsguide` en ligne de commande.

Bibliographie

- C. Rolland, *LaTeX par la pratique*, éd. O'Reilly (1999), ISBN 2-84177-073-7, p. 353–380

Programmer avec LaTeX

Il est possible de mettre du code TeX dans un fichier LaTeX ; le langage TeX contient des structures de programmation (voir *Programmation TeX*). Mais il est aussi possible de programmer avec du LaTeX, moyennant l'utilisation d'extensions spécifiques.

Cela permet de faire des commandes personnelles (macros) à comportement « variable », par exemple en calculant la largeur et la hauteur du texte passé en paramètre.

Variables

Il existe trois manières de créer des variables en LaTeX : avec des commandes, des longueurs ou des compteurs. Dans tous les cas, le nom de la variable suit le formalisme des commandes personnelles : il commence par une contre-oblique et ne doit contenir que des lettres.

L'avantage des longueurs et des compteurs est qu'ils ont des commandes dédiées pour les manipuler (définition, addition, utilisation, …).

Commandes

Les commandes personnelles sont des variables de base. Par exemple, on peut définir

```
\usepackage{graphicx}
...
\newcommand{\echelle}{0.125}

\includegraphics[scale=\echelle]{image1}
\includegraphics[scale=\echelle]{image2}
\includegraphics[scale=\echelle]{image3}
```

Longueurs

Une longueur doit contenir un nombre, entier ou décimal (le séparateur étant le point) et une unité accolée (cf. *Éléments de base > Espaces et changements de ligne*). La déclaration d'une longueur se fait avec la commande `\newlength`, et l'assignation de la valeur se fait avec `\setlength`, par exemple :

```
\newlength{\malongueur}
\setlength{\malongueur}{1.5em}
```

crée une longueur `\malongueur` et lui donne la valeur d'un cadratin et demi. Autre exemple :

```
\usepackage{graphicx}
...
\newlength{\largeur}{0.125}
\setlength{\largeur}{10cm}

\includegraphics[width=\largeur]{image1}
\includegraphics[width=\largeur]{image2}
\includegraphics[width=\largeur]{image3}
```

On peut assigner la longueur d'un objet, d'un mot, avec `\settowidth` :

```
\newlength{\malongueur}
\settowidth{\malongueur}{Bonjour}
```

la longueur `\malongueur` aura pour valeur la taille du mot « Bonjour » dans la police courante. La longueur s'utilise ensuite à la place des valeurs dans les commandes, par exemple :

```
\hspace{\malongueur}
```

On peut ajouter une valeur à une longueur avec `\addtolength` :

```
\addtolength{\malongueur}{1em}
```

augmente la longueur `\malongueur` d'un cadratin.

Voir aussi *LaTeX/Mise en forme du texte (avancé)#Déformation du texte*.

LaTeX possède des longueurs prédéfinies qui ajustent la mise en forme du texte. Pour ce qui est des marges et de l'interligne, on préfèrera faire confiance aux extensions dédiées `geometry` (cf. *Mise en page*), `fancyhdr` et `setspace` (cf. *Mise en forme du texte (avancé) > Espacement*

interligne). On peut avoir la liste des longueurs de mise en page avec l'extension `[francais]{layout}`, qui définit une commande `\layout` affichant une page avec la définition des longueurs.

Voici quelques longueurs utiles :

- paragraphes :
  - `\parindent` : alinéa (retrait de paragraphe),
  - `\parskip` : espacement entre les paragraphes,
  - `\baselineskip` : interligne ;
- tableaux :
  - `\tabcolsep` : moitié de la longueur séparant deux colonnes,
  - `\arrayrulewidth` : largeur des filets,
  - `\doublerulesep` : espacement entre deux filets pour un filet double ;
- cadres :
  - `\fboxrule` : épaisseur du filet,
  - `\fboxsep` : espacement entre le cadre et le contenu.

Compteurs

Un compteur est une variable entière. Elle est créée par la commande `\newcounter` et prend la valeur nulle. On peut lui attribuer une valeur autre avec `\setcounter` :

```
\newcounter{moncompteur}
\setcounter{moncompteur}{4}
```

On peut afficher cette valeur avec la commande `\the`:

```
\themoncompteur
```

On peut l'incrémenter avec la commande `\addtocounter` :

```
\addtocounter{moncompteur}{2}
```

ajoute 2 au compteur `moncompteur`. Les compteurs sont utilisés par LaTeX pour les listes ordonnées et les numérotations de chapitres, sections, sous-sections, …

Extensions spécifiques

Extension `calc`

L'extension `calc` permet d'effectuer des calculs arithmétiques sur les compteurs et les longueurs (par défaut, on ne peut que multiplier des valeurs dans LaTeX). Par exemple, pour ajouter deux compteurs « cA = cB + cC » on peut écrire

```
\usepackage{calc}
...
\newcounter{cA}
\newcounter{cB}
\newcounter{cC}
...
\setcounter{cA}{\value{cB} + \value{cC}}
```

au lieu de

```
\newcounter{cA}
\newcounter{cB}
\newcounter{cC}
...
\setcounter{cA}{\value{cB}}
\addtocounter{cA}{\value{cC}}
```

Par ailleurs, l'extension permet aussi de multiplier et de diviser avec les symboles standards `*` et `/`. Il existe toutefois quelques restrictions ; en particulier, on ne peut — évidemment — pas ajouter un compteur avec une longueur, et un compteur contient toujours un nombre entier (si le résultat est décimal, il est tronqué). Par ailleurs, si l'on veut multiplier une longueur par un nombre, il faut mettre la longueur en premier : `1em*2`.

Extension `multido`

L'extension `multido` permet de faire une boucle itérative incrémentale.

Exemple

```
\usepackage{multido}
...
\multido{\i=1+1}{50}{%
  la valeur est \i{}
}
```

Extension `ifthen`

L'extension `ifthen` fournit des structures de contrôle : exécutions conditionnelles et boucles conditionnelles.

Mettre du code interprété par le visualiseur

La commande `\special{`*`code`*`}` permet de mettre du code qui sera écrit tel quel dans le fichier DVI. Il sera ainsi interprété et exécuté par le visualiseur DVI.

La commande `\pdfcatalog{`*`code`*`}` a le même rôle pour les fichiers PDF.

Voir aussi

Bibliographie

- D. Bitouzé, J.-C. Charpentier, *LaTeX — Synthèse de cours & exercices*, éd. Pearson Education, ISBN 2-7440-7187-0, p. 264–270, 273, 276–278
- C. Rolland, *LaTeX par la pratique*, éd. O'Reilly (1999), ISBN 2-84177-073-7, p. 141–148, 341–342, 354–355

Vadémécum

Comme précisé en introduction, vous n'avez besoin la plupart du temps de ne connaître que peu d'instructions. Si l'on reprend la loi de Pareto, parmi toutes les commandes que vous utiliserez, 80 % de vos documents n'utiliseront que 20 % de ces commandes.

Par exemple, si l'on se contente de faire uniquement du texte, soit 80 % des besoins en général (si l'on excepte les ouvrages contenant beaucoup de mathématiques), on a besoin de connaître au plus une dizaine d'instruction (`\chapter`, `\section`, `\subsection`, `\emph`, `\footnote`, `\label`, `\ref`, `\pageref`) et d'environnements (`itemize`, `enumerate`, `quote`, `quotation`) — les commandes de début et de fin de document étant dans des modèles. Si l'on veut aller un peu plus loin, une vingtaine de commandes et environnements suffisent (tableaux, images). Ce sont ces 20 % de commandes.

Ce document présente également les commandes supplémentaires les plus courantes, en espérant couvrir 96 % de vos besoins… C'est donc une sorte de *vade-mecum*, un document à garder sur vous (ou sur votre disque dur) comme pense-bête.

Ce sont les commandes que nous avons vues précédemment, mais présentées succinctement, afin d'être utilisé comme aide-mémoire.

> **Remarque :** pensez à indiquer vos modifications dans la section « À faire » de la page de *commons*.

Syntaxe de base

Les commandes, ou instructions, commencent par une contre-oblique `\`.

Certaines commandes s'utilisent sans paramètre. Si ces commandes sont formées de lettres, l'espace qui les suit est ignoré : il indique que c'est la fin de la commande, mais ne provoque pas d'espacement. En cas de problème, on peut ajouter un bloc vide, par exemple `\^\i{}`, `\fg{}`, `\copyright{}`, …

Certaines commandes admettent un ou plusieurs paramètres. Les paramètres optionnels sont entre crochets `[…]`, on peut mettre plusieurs paramètres séparés par une virgule. Les paramètres obligatoires sont entre accolades `{…}`, et on ne met qu'un paramètre par accolade.

Un environnement commence par `\begin{`*`nom_env`*`}` et se termine par `\end{`*`nom_env`*`}`.

Les dimensions (longueurs, largeurs) sont indiquées sous la forme d'un nombre et d'une unité accolée. Par exemple, `3cm` désigne trois centimètres, `1em` désigne un cadratin. Les unités sont : `mm` (millimètre), `cm` (centimètre), `pt` (point anglo-saxon), `dd` (point Didot), `ex` (hauteur d'x) et `em` (cadratin).

On peut également utiliser des longueurs prédéfinies : `\linewidth` (largeur du texte, justification), `\baselineskip` (distance entre la ligne de base de deux lignes consécutives d'un même paragraphe), `\parindent` (largeur du retrait de paragraphe) et `\parskip` (interligne entre deux paragraphes).

Exemple

```
\hspace{4em}
\vspace{0.5\baselineskip}
```

Certaines commandes, dites « fragiles », ne donnent pas le résultat attendu dans certains environnement ou arguments de commandes. Le fait de placer devant la commande `\protect` peut parfois résoudre le problème.

Fichier de préambule

Le fichier de préambule est un fichier `.tex` qui contient :

- les appels aux extensions *(packages)* ;
- les commandes personnelles.

On l'appellera ici `preambule.tex`

Voici un fichier de préambule typique :

```
% ************************
% * fichier de préambule *
% ************************

% ***** extensions *****

\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{graphicx}
\usepackage[frenchb]{babel}

% ***** césures particulières *****
```

```
\hyphenation{anti-consti-tu-tionnel-le-ment atmo-sphère
   caou-tchouc cis-alpin trans-action}

% ***** exemples de commandes personnelles *****

\newcommand{\langue}{\emph} % mots en langues étrangères
\newcommand{\citital}{\emph} % citation en italique
\newcommand{\nomprog}{\texttt} % nom de programme en police teletype
```

On peut de même créer un fichier de fin de document, par exemple pour un livre, le fichier `fin_livre.tex` :

```
% ******************
% * fichier de fin *
% ******************
\listoftables
\listoffigures
\bibliographystyle{} % indiquer le style
\bibliography{} % indiquer le fichier .bib

\tableofcontents
```

Commandes personnelles

Pour une commande sans paramètre (par exemple développer une abréviation) :

```
\newcommand{\nom_de_commande}{commandes à exécuter}
```

Pour une commande avec *n* paramètres :

```
\newcommand{\nom_de_commande}[n]{commandes à exécuter}
```

les paramètres figurant sous la forme `#1`, `#2`, …

Pour définir un nouvel environnement :

```
\newenvironment{nom_de_l_environnement}
  {commandes de début}
  {commandes de fin}
```

Squelette de fichier

Le squelette de fichier est un fichier minimal qui sert de base à tous les autres fichiers. Pour l'utiliser, on fait une copie du fichier, on lui donne le nom du document puis on le modifie.

Le squelette général est :

```
\documentclass[options]{classe}

\input{preambule}

\begin{document}

\end{document}
```

On peut créer des squelettes pour différents types de document. Pour un livre, on aura par exemple

```
\documentclass[a4paper, 11pt]{book}

\input{preambule}

\begin{document}

\title{}
\author{}
\date{}

\frontmatter % introduction

\maketitle

\mainmatter % corps du livre
```

```
\include{} % fichier contenant le 1er chapitre
\include{} % fichier contenant le 2e chapitre, ...

\appendix

\include{} % fichier contenant la 1re annexe

\backmatter % annexes

\input{fin_livre}

\end{document}
```

et pour un article

```
\documentclass[a4paper, 11pt]{article}

\input{preambule}

\begin{document}

\title{}
\author{}
\date{}

\maketitle

\begin{abstract}
   % résumé de l'article

\end{abstract}

% corps de l'article

\end{document}
```

Notons que :

- le titre dans `\title` est nécessairement un paragraphe unique ; on peut forcer un retour de ligne avec `\\` ;
- dans `\author`, le nom d'un auteur peut être suivi d'une instruction `\thanks{…}`, qui permet de lui associer une note de bas de page ;
- la fonction `\include` permet d'utiliser le texte d'un fichier `.tex`, ce qui permet de morceler un grand document en plusieurs fichiers ; la fonction introduit également une nouvelle page ;
- la fonction `\input` est similaire à `\include` mais ne commence pas une nouvelle page, ce qui est utile pour les articles par exemple ; un fichier appelé par un `\input` ou un `\include` peut lui-même contenir un `\input` mais pas un `\include`.

Format du document

Le format du document dépend essentiellement de la classe de document invoquée : `book` (livre), `report` (rapport), `article` (article) pour les principales. On peut indiquer le format du papier ainsi que son orientation :

```
\documentclass[a4paper,11pt]{article}
```

pour un format A4 avec un corps de texte de 11 points. Les principales options sont (les options sur une même ligne sont exclusives) :

- `10pt`, `11pt`, `12pt` : corps de texte ;
- `a4paper`, `a5paper`, `b5paper`, `legalpaper`, `letterpaper` : format de la page ;
- `landscape` : page orientée à l'italienne (en paysage) ;
- `twocolumn` : texte sur deux colonnes.

Par défaut, la page est orientée en portrait, et le texte est sur une colonne.

On peut définir le format avec les extensions `geometry` et `multicol` (qui permet d'utiliser plus de deux colonnes), auquel cas on n'indique que le corps de texte avec la classe :

```
\documentclass[11pt]{article}

\usepackage[a4paper,landscape]{geometry}
\usepackage{multicol}

\begin{document}
\begin{multicols}{2)
...
\end{multicols}
\end{document}
```

Structuration du document

Titres de parties

Titres de parties

Type de partie	Commande	Remarque
Partie	`\part{`*`titre`*`}`	
Chapitre	`\chapter{`*`titre`*`}`	n'est pas disponible pour la classe `article`
Section	`\section{`*`titre`*`}`	
Sous-section	`\subsection{`*`titre`*`}`	
Sous-sous-section	`\subsubsection{`*`titre`*`}`	
Paragraphe	`\paragraph{`*`titre`*`}`	
Sous-paragraphe	`\subparagraph{`*`titre`*`}`	

Le titre affiché dans la table des matières peut être différent de celui affiché sur la page :

```
\section[titre dans la table]{Titre réel}
```

On peut utiliser les commandes étoilées (par exemple `\chapter*{`*`titre`*`}`) si l'on ne veut pas que le titre soit numéroté ; il ne figure alors pas dans la table des matières.

Si l'on veut au contraire faire figurer une partie, on utilise `\addcontentsline`, par exemple :

```
\addcontentsline{toc}{section}{\protect\numberline{}Préface}
\chapter*{Préface}
```

Le deuxième argument est le niveau de titre (`chapter`, `section`, `subsection`, …). La bibliographie, l'index, les listes de tableau et de figure et la table des matière elle-même peuvent être inclus automatiquement (sans utiliser `\addcontentsline`) en utilisant l'extension `tocbibind`.

Si vous utilisez l'extension `hyperref`, les entrées de la table des matières seront des liens hypertexte vers les sections.

Mise en emphase

La mise en emphase se fait avec

```
\emph{texte}
```

Cela se traduit par de l'italique dans le romain, ou par du romain dans l'italique.

On peut aussi mettre du texte hors alinéa, avec une marge plus grande. On utilise pour cela :

- l'environnement `quote` : en général pour les citations courtes, la composition est en pavé ;
- l'environnement `quotation` : en général pour les citations longues, la composition est en alinéa et il doit contenir plusieurs paragraphes.

Listes structurées

Les listes structurées sont des environnements :

```
\begin{type_de_liste}
   \item premier objet de la liste~;
   \item deuxième objet de la liste~;
   …
\end{type_de_liste}
```

Types de liste

Type de liste	Environnement
Liste numérotée	`enumerate`
Liste non-numérotée	`itemize`
Liste de description	`description`

Les listes de description s'utilisent de la manière suivante

```
\begin{description}
   \item[objet 1] description de l'objet 1~;
```

```
   \item[objet 2] description de l'objet 2~;
   …
\end{description}
```

On peut imbriquer les listes

```
\begin{type_de_liste 1}
   \item premier objet de la liste~;
   \item deuxième objet de la liste~:
      \begin{type_de_liste 2}
         \item premier objet de la sous-liste,
         …
      \end{type_de_liste 2}
   …
\end{type_de_liste 1}
```

Références et notes

Note

Une note de bas de page s'obtient avec

```
\footnote{texte de la note}
```

placé à l'endroit de l'appel de note.

Si vous utilisez l'extension `hyperref`, les appels de note sont des liens vers la note.

La commande `\marginpar{`*texte de la note*`}` permet de mettre une note de marge, mais sans appel de note (puisque la note se trouve à côté du texte auquel elle se réfère).

Référence croisée

Pour faire une référence dans le texte (à une page, à un numéro de section), il faut d'abord placer une étiquette à l'endroit cible :

```
\label{étiquette}
```

puis, on utilise

- `\pageref{`*étiquette*`}` pour mentionner le numéro de page, et
- `\ref{`*étiquette*`}` pour mentionner le numéro de partie, chapitre, section, sous-section, équation, objet flottant (figure, tableau).

Si vous utilisez l'extension `hyperref`, les numéros de section et de page sont des liens menant vers l'endroit pointé.

Bibliographie

Pour placer une référence bibliographique, il faut :

- introduire les références de l'ouvrage dans un fichier `.bib` dédié ; pour chaque ouvrage, on définit une étiquette ;
- à l'endroit de l'appel, écrire `\cite{`*étiquette*`}` ;
- à la fin de l'ouvrage, mettre
  `\bibliographystyle{`*style*`}`
  `\bibliography{`*nom de fichier*`}` ;
- compiler avec `latex`, puis avec `bibtex`, et à nouveau avec `latex`.

Les entrées du fichier de bibliographie sont du type

```
@book{étiquette,
   author="auteur(s) du livre",
   title="titre de l'ouvrage",
   year="année",
   publisher="éditeur"
}
```

pour les livres, et

```
@article{étiquette,
   author="auteur(s) de l'article",
   title="titre de l'article",
   journal="nom du journal",
```

```
   number="numéro du journal",
   year="année de parution"
}
```

Il ne faut employer que des caractères ASCII (non accentués), donc utiliser des commandes de type `{\'e}` pour « é ». Les mots contenant des capitales doivent êtres dans un bloc `{…}` pour que la casse soit respectée.

S'il n'y a qu'un seul auteur on écrit :

```
author="Nom, Prénom"
```

s'ils sont plusieurs, on écrit :

```
author="Nom1, Prénom1 and Nom2, Prénom2"
```

Pour les styles bibliographiques, on peut utiliser des styles anglais ou francisés. Les principaux styles sont :

- `plain` (anglais) ou `plain-fr` (francisé) : style « simple », la référence est un numéro, établi selon l'ordre alphabétique des auteurs puis l'année ;
- `unsrt` (anglais) ou `unsrt-fr` (francisé) : style « non-trié » *(unsorted)*, la référence est un numéro qui est l'ordre de citation ;
- `alpha` (anglais) ou `alpha-fr` (francisé) : style « alphanumérique », la référence est composée des initiales du ou des auteurs et de l'année.

Si vous utilisez l'extension `hyperref`, les appels de note bibliographique sont des liens vers la référence.

Index

Pour faire un index, il faut utiliser l'extension `makeidx`, puis :

- dans le préambule, mettre la commande `\makeindex` ;
- pour créer une entrée, mettre dans le texte `\index{`*`nom de l'entrée`*`}` ;
- mettre la commande `\printindex` à l'endroit où l'on veut mettre l'index ;
- compiler avec `\latex`, puis avec `makeindex`, puis à nouveau avec `latex`.

L'index contiendra le *nom de l'entrée* suivi du, ou des numéros de page.

On a les possibilités suivantes :

- pour avoir un classement alphabétique ne correspondant pas au nom de l'entrée : `\index{`*`nom de classement@nom affiché`*`}`, par exemple `\index{Epee@Épée}` ;
- pour faire référence à un autre mot : `\index{`*`mot`*`|see{`*`autre mot`*`}}`, par exemple `\index{Sabre|see{Épée}}` ;
- pour avoir un intervalle de page : on place `\index{`*`mot`*`|(}` au début de la zone et `\index{`*`mot`*`|)}` à la fin ;
- pour une « cascade », on met un point d'exclamation, par exemple `\index{Lame!Sabre}`, `\index{Lame!Epee@Épée}`.

Si vous utilisez l'extension `hyperref`, les numéros de page sont des liens vers les endroits concernés.

Caractères particuliers

Certains caractères ne sont pas accessibles au clavier, ou bien ne font pas partie du codage utilisé (défini par l'option passée à l'extension `inputenc`). Pour les obtenir, on utilise des commandes. En particulier, les caractères suivants ont une utilisation spéciale dans LaTeX et ne peuvent être tapés tels quels : `{ } % # $ ^ ~ & _ \`.

Nous indiquons ci-dessous quelques caractères spéciaux. L'extension `textcomp` fournit des caractères supplémentaires.

Caractères spéciaux

Saisie	Caractère	Saisie	Caractère	Saisie	Caractère
`\{`	{	`\copyright`	©	`\textbar`	|
`\}`	}	`\dag`	†	`\textperiodcentered`	·
`\%`	%	`\ddag`	‡	`\textregistered`	®
`\#`	#	`\o`	ø	`\texttrademark`	™
`\$`	$	`\O`	Ø	`\textvisiblespace`	␣
`\textasciicircum`	^	`\P`	¶	`\degres`?[2]	°
`\textasciitilde`	~	`\pounds`	£	`\ier`, `\iere`, `\iers`, `\ieres`?[2]	er, re, ers, res
`\&`	&	`\S`	§	`\ieme`, `\iemes`?[2]	e, es
`\_`	_	`\ss`	ß	`\primo`, `\secundo`, `\tertio`, `\quarto`[2]	1°, 2°, 3°, 4°
`\textbackslash`	\	`\euro`?[1]	€	`\no`, `\No`[2]	n°, N°

Notes

1. avec l'extension `eurosym`
2. avec l'extension `[frenchb]{babel}`

Ponctuation particulière

Commande	Caractère	Note
`\dots`	…	points de suspension
`\og`, `\fg`	«, »	avec l'extension `[frenchb]{babel}` ; gère les espaces insécables ; selon ce qui suit le guillemet fermant, on peut devoir utiliser `\fg{}`
`-`	-	division (trait d'union)
`--`	–	tiret demi-cadratin, alternative : `\endash`
`---`	—	tiret cadratin, alternative : `\emdash`
`?` `	¿	alternative : `\textquestiondown`
`!` `	¡	alternative : `\textexclamdown`

Diacritiques et ligatures

Commande	Résultat	Exemple
`\'`	accent aigu	`\'E` ? É
`\` `	accent grave	`\` `E` ? È
`\^`	accent circonflexe	`\^E` ? Ê, `\^\i` ? î
`\"`	tréma	`\"E` ? Ë
`\c`	cédille	`\c?C` ou `\c{C}` ? Ç
`\ae`, `\AE`, `\oe`, `\OE`	æ, Æ, œ, Œ	

On peut utiliser différents espaces horizontaux. Pour introduire une espace[1] de dimension donnée, on utilise la commande

```
\hspace{longueur}
```

Espaces

Type	Saisie
espace justifiante	?, retour de ligne, `\?`, `{}?` ou `{?}`
espace insécable	~
petite espace	`\,`
espace fine	`\/`

Rappel : l'espace insécable se met entre autres devant une ponctuation double (:;?!), l'extension `[frenchb]{babel}` gère ceci automatiquement ; elle se met aussi entre une valeur et son unité, ainsi qu'entre un titre et un patronyme.

Mise en forme du texte

Police

On peut faire varier le corps (taille du texte), par ordre de taille croissant : `\tiny`, `\scriptsize`, `\footnotesize`, `\small`, `\normalsize` (taille normale), `\large`, `\Large`, `\LARGE`, `\huge`, `\Huge`. On peut :

- soit inclure le texte avec la commande dans un bloc, par exemple `{\small` *`texte en petit`*`}` ;
- soit utiliser un l'environnement ayant le même nom que la commande, par exemple `\begin{small}` *`texte en petit`* `\end{small}`.

On peut changer la police. Une police est définie par trois paramètres :

- sa famille (caractères romains, sans empattement, type machine à écrire) ;
- sa forme (caractères droits, italiques, petites capitales) ;
- sa graisse (gras, moyen).

On peut définir chacun de ces trois paramètres avec des commandes à argument, de la forme `\text`*`option`*, ou bien avec des commandes déclaratives ; on a trois type de commandes déclaratives selon le paramètre concerné (`\`*`option`*`family` pour la famille, `\`*`option`*`shape` pour la forme, `\`*`option`*`series` pour la graisse).

Choix de la police

Type de police	Commande à argument	Commande déclarative	Remarque
romain	`\textrm{`*`texte`*`}`	`{\rmfamily` *`texte`*`}`	
sans empattement	`\textsf{`*`texte`*`}`	`{\sffamily` *`texte`*`}`	
police type machine à écrire	`\texttt{`*`texte`*`}`	`{\ttfamily` *`texte`*`}`	
italique	`\textit{`*`texte`*`}`	`{\itshape` *`texte`*`}`	la plupart du temps, préférer `\emph{…}`
petites capitales	`\textsc{`*`texte`*`}`	`{\scshape` *`texte`*`}`	
	`\bsc{`*`texte`*`}`		avec l'extension `[frenchb]{babel}` (empêche la césure)
gras	`\textbf{`*`texte`*`}`	`{\bfseries` *`texte`*`}`	
supérieures (exposant)	`\textsuperscript{`*`texte`*`}`		
chiffres elzéviriens (ou bas de casse)	`\oldstylenums{`*`chiffres`*`}`		

Certaines extensions permettent d'utiliser d'autres fontes :

- l'extension `fourier` permet d'utiliser la fonte Utopia ;
- l'extension `mathptmx` permet d'utiliser la fonte Times ;
- l'extension `mathpazo` permet d'utiliser la fonte Palatino ;
- l'extension `times` définit la fonte Times comme fonte romaine, Helvetica comme fonte sans empattement et `Courier` comme fonte à chasse fixe ;
- l'extension `oldgerm` fournit la commande `\textgoth{`*`texte en gothique`*`}` ; l'extension `yfonts` fournit les commandes `{\frakfamily` *`texte en gothique`*`}` et `\textfrak{`*`texte en gothique`*`}` ;
- l'extension `frcursive` fournit l'environnement `cursive`, qui imite une écriture manuelle.

Pour une lettrine, il faut utiliser l'extension `lettrine`, on a alors la commande `\lettrine` :

```
\lettrine{L}{es premiers} mots du paragraphe...
```

Mise en page

On peut provoquer un retour à la ligne sans changer d'alinéa avec `\\` (double contre-oblique).

On peut introduire un espace vertical entre deux paragraphes :

- `\vspace{`*`longueur`*`}` ;
- `\medskip` pour sauter une ligne « normale » ;
- `\smallskip` pour un « petit » saut de ligne ;
- `\bigskip` pour un « grand » saut de ligne.

On peut inciter à un saut de page avec `\pagebreak` et décourager un saut de page avec `\nopagebreak`.

On peut forcer un saut de page avec `\newpage` ou `\clearpage`, le second forçant l'affichage des flottants en suspens. Avec la commande `\cleardoublepage`, le texte qui suit est en belle page (page impaire).

L'environnement `minipage` permet d'organiser une partie d'un texte comme une page, par exemple à l'intérieur d'une `\fbox`, ou bien pour placer des notes au bas d'un bloc donné plutôt qu'en bas de page :

```
\begin{minipage}{largeur}
```

```
   ...
\end{minipage}
```

Composition

Par défaut, le texte est composé en alinéa en typographie française.

Pour composer ponctuellement en pavé, on peut mettre un espace vertical (par exemple `\medskip`) avant le paragraphe pour placer une ligne vide, et `\noindent` au début du paragraphe pour annuler l'alinéa.

La commande `\indent` permet de forcer un alinéa lorsqu'il n'y en a pas.

Pour une composition en drapeau (alignée à gauche ou à droite), on utilise les environnements `flushleft` (au fer à gauche) ou `flushright` (au fer à droite). Pour une composition centrée, on utilise l'environnement `center`.

Pour empêcher une césure, on utilise `\mbox{`*`mot`*`}`. Pour indiquer l'endroit où peut se trouver la césure, on utilise la commande `\hyphenation` dans le préambule. S'il s'agit d'un mot à usage unique, on peut indiquer la césure localement, en mettant `\-` dans le mot.

Tableaux

Il est fortement conseillé d'utiliser l'extension `array`.

Un tableau est défini par l'environnement `tabular` :

```
\begin{tabular}{description}
   élément A1 & élément A2 \\
   élément B1 & élément B2 \\
\end{tabular}
```

Les lignes se terminent par une double contre-oblique `\\` ou par un `\tabularnewline`, et au sein d'une ligne, les cellules sont séparées par une esperluette `&`.

La *description* est la liste des colonnes avec leur type :

- `l` : texte aligné à gauche ;
- `c` : texte centré ;
- `r` : texte aligné à droite ;
- `p{`*`largeur`*`}` : texte en pavé, la largeur de la colonne étant fixée.

Dans une colonne de type `p`, on peut revenir à la ligne au sein d'une colonne avec `\newline`, et on peut changer d'alinéa avec une ligne vide.

Pour avoir des filets verticaux, on place des tubes `|` dans la *description*. Pour avoir des filets horizontaux, on place des `\hline`. On peut mettre un filet horizontal partiel (qui ne fait pas toute la largeur du tableau) avec `\cline{`*`colonne1-colonne2`*`}`. L'extension `arydshln` (pour *array dashed lines*, à placer après les autres extensions relatives aux tableaux) qui permet d'avoir un filet en trait discontinu : on utilise le deux-points « : » pour les séparateurs verticaux, `\hdashline` pour les séparateurs horizontaux et `cdashline{`*`colonne1-colonne2`*`}` pour les séparateurs horizontaux qui ne font pas toute la largeur. Par exemple, pour un tableau avec des filets simples partout :

```
\begin{tabular}{|l|l|}
   \hline
   élément A1 & élément A2 \\
   \hline
   élément B1 & élément B2 \\
   \hline
\end{tabular}
```

On peut fusionner des cellules consécutives d'une ligne avec `\multicolumn` :

```
\multicolumn{nombre}{description}{contenu}
```

où *nombre* est le nombre de cellules fusionnées, et *description* est la description comme ci-dessus (`l`, `c`, `r`, `p`). À moins que la première colonne ne soit incluse, on n'indique que le filet de droite (par exemple `{l|}`), le filet de gauche étant déterminé par la description générale. On peut utiliser cette commande sur une seule colonne pour en changer la mise en forme (alignement, filets).

On peut simuler une fusion verticale en utilisant des filets partiels. Pour faire de la vraie fusion de lignes, il faut utiliser l'extension `multirow` et la commande :

```
\multirow{nombre de lignes}{largeur}{contenu}
```

Pour *largeur*, on peut indiquer une étoile `*` afin de laisser LaTeX ajuster la largeur automatiquement ; cependant, cette option ne permet pas d'avoir

des retours de ligne dans la cellule.

L'extension `array` définit deux commandes à placer dans la description des colonnes : `>{`*commandes*`}` qui permet d'exécuter des commande en début de cellule, et `<{`*commandes*`}` pour la fin de cellule. Par exemple :

- mettre toute une colonne en fonte teletype : `>{\ttfamily}l|` ;
- mettre toute une colonne en mode mathématiques : `>{$}l<{$}|` ;
- pour aligner gauche en fixant la largeur : `>{\raggedright\arraybackslash}p{3cm}|`[2] ; la commande `\arraybackslash` permet d'utiliser `\\` pour changer de ligne de tableau, sinon ceci sert pour revenir à la ligne dans la cellule (on peut toujours utiliser `\tabularnewline`) ;
- pour centrer et aligner à droite : `>{\centering}` et `>{\raggedleft}`, avec la même précaution que ci-dessus.

Cette extension fournit également deux types de colonne : `m` et `b`, identiques à `p`, mais verticalement, les cellules *des autres colonnes* seront respectivement centrées et alignées en bas par rapport aux cellules de cette colonne. Si l'on veut centrer verticalement une cellule ou l'aligner en bas, il faut avoir recours à une solution au cas par cas[3].

Enfin, si un type de colonne doit être utilisé plusieurs fois, l'extension `array` permet de définir un type de colonne, par exemple

```
\newcolumntype{C}{>{\centering\arraybackslash}p{3cm}}
\begin{tabular}{|l|C|}
   ...
```

L'extension `slashbox` permet de scinder une cellule en deux selon la diagonale, par exemple pour la cellule en haut à gauche du tableau, on utilise :

```
\backslashbox{titre de la colonne}{titre de la ligne}
```

Si l'on veut faire tourner un texte (par exemple mettre un en-tête vertical), on utilise

```
\rotatebox{angle}{texte}
```

où *angle* est en degrés. Si l'on veut avoir un en-tête de colonne à 45°, il faut mettre le texte dans une boîte de taille nulle pour ne pas élargir la colonne :

```
\makebox[0cm][1]{\rotatebox{45}{texte}}
```

Couleur

On utilise l'extension `xcolor`. On dispose des fonctions suivantes :

- `\textcolor{`*couleur*`}{`*texte*`}` et `{\color{`*couleur*`} `*texte*`}` pour la couleur des caractères ;
- `\colorbox{`*couleur*`}{`*texte*`}` pour une boîte à fond coloré ;
- `\fcolorbox{`*couleur cadre*`}{`*couleur fond*`}{`*texte*`}` pour une boîte à fond coloré avec un filet de couleur autour du texte ;
- `\pagecolor{couleur}` pour une page à fond coloré.

Les couleurs utilisables sont `red`, `green`, `blue`, `cyan`, `magenta`, `yellow`, `orange`, `violet`, `purple`, `brown`, `black`, `darkgray`, `gray`, `lightgray` et `white`.

On peut aussi utiliser des modèles, en remplaçant `{`*couleur*`}` par `[`*modèle*`]{`*couleur*`}`, par exemple `\colorbox[`*modèle*`]{`*couleur*`}{`*texte*`}`. On a notamment :

- `rgb` : les trois composantes sont un nombre entre 0 et 1, séparés par une virgule ;
- `HTML` : les trois composantes sont un nombre hexadecimal entre 00 et FF, accolés (HTML en majuscule) ;
- `hsb` : modèle teinte-saturation-luminosité (TSL), les trois composantes sont un nombre entre 0 et 1 (la teinte est donc l'angle en degrés divisé par 360), séparés par une virgule ;
- `gray` : modèle nuances de gris, avec un nombre décimal entre 0 (noir) et 1 (blanc).

On peut définir une nouvelle couleur avec `\definecolor` :

```
\definecolor{vertolive}{rgb}{0.5,0.5,0}
```

Pour un tableau, on utilise le paramètre `table` en appelant l'extension `xcolor`, `\usepackage[table]{xcolor}`. On peut alors définir :

- la couleur de fond d'une ligne en mettant `\rowcolor{`*couleur*`}` en début de ligne ;
- la couleur de fond d'une colonne en mettant `>{\columncolor{`*couleur*`}}` avant la désignation de la colonne dans la définition de l'environnement `tabular` ; on peut lui adjoindre la couleur des caractères `>{\color{`*couleur*`} \columncolor{`*couleur*`}}` ;
- la couleur de fond d'une cellule en mettant `\cellcolor{`*couleur*`}` en début de cellule.

```
\begin{tabular}{|l|l|>{\columncolor{yellow}}l|}
```

```
   \hline
   A1 & A2 & A3 \\ \hline
   \rowcolor{cyan} B1 & B2 & \cellcolor{green}B3 \\ \hline
\end{tabular}
```

Pour une alternance de couleur sur deux lignes, on peut utiliser la commande `\rowcolors` (avec un *s*?) : `\rowcolors{`*`début`*`}{`*`couleur impaire`*`}{`*`couleur paire`*`}`, *début* étant la première ligne colorée (cela permet de sauter les en-têtes).

```
\rowcolors{1}{lightgray}{white}
\begin{tabular}
...
```

Éléments autres que du texte

Images

L'inclusion d'une image se fait avec la commande `\includegraphics{`*`nom du fichier`*`}`. Il est conseillé d'avoir deux fichiers image, un PostScript (`.ps` ou `.eps`) et un PNG, JPEG ou PDF, et de ne pas indiquer l'extension afin de laisser LaTeX choisir le fichier.

L'extension `graphicx` permet de définir simplement la taille de l'image : `\includegraphics[width=`*`largeur`*`]{`*`nom du fichier`*`}`, `\includegraphics[height=`*`hauteur`*`]{`*`nom du fichier`*`}`.

Adresses réticulaires

L'extension `hyperref` met en forme l'adresse réticulaire et crée en plus un lien cliquable dans le document :

```
\url{http://fr.wikibooks.org/}
\url{mailto:jeveuxduspam@fai.fr}
\url{news:fr.comp.text.tex}
```

Comme l'extension redéfinit un certain nombre de commandes, il est recommandé de la placer en dernier dans le préambule. Si l'on veut colorer le lien, on peut appeler l'extension avec l'option `colorlinks=true`. L'option `breaklinks=true` autorise LaTeX à faire des césures dans les adresses :

```
\usepackage[colorlinks=true,breaklinks=true]{hyperref}
```

Si vous ne voulez pas de lien mais simplement une mise en forme, vous pouvez utiliser la commande `\url{…}` de l'extension `url`.

Éléments flottants

Un élément flottant est une image ou un tableau dont le placement est laissé à l'initiative de LaTeX ; LaTeX se charge également de la numérotation. Un élément flottant est défini par un environnement : `figure` pour une image et `table` pour un tableau.

```
\begin{figure}[position]
   \caption{\label{étiquette} titre}
   \includegraphics[…]{…}
\end{figure}
[…]
Dans la figure~\ref{étiquette} page~\pageref{étiquette}, […].
```

ou bien

```
\begin{table}[position]
   \caption{\label{étiquette} titre}
   \begin{tabular}{description}
      […]
   \end{tabular}
\end{table}
[…]
Dans le tableau~\ref{''étiquette''} page~\pageref{''étiquette''}, […].
```

Le paramètre optionnel *position* est une lettre indiquant l'emplacement désiré :

- `h` pour qu'il soit à côté du texte précédant dans le source *(here)*,
- `t` : en haut d'une page *(top)*,
- `b` : en bas d'une page *(bottom)*,
- `p` : dans une page ne contenant que des flottants (regroupement des figures et tableaux).

Si l'on est sur deux colonnes, les environnements étoilés `figure*` et `table*` permettent de placer l'objet sur toute la largeur de la page et non plus dans une colonne.

Pour centrer l'objet, il faut utiliser `\centering` :

```
\begin{figure}[position]  \begin{table}[position]
   \centering                \centering
   […]                          […]
\end{figure}              \end{table}
```

La commande `\clearpage` qui provoque un changement de page et l'affichage de tous les flottants en attente. L'instruction `\cleardoublepage` a le même effet, mais le texte qui suit est placé en belle page.

On peut afficher l'index des flottants avec `\listoffigures` et `\listoftables`.

L'extension `rotating` fournit deux environnements flottants appelés `sidewaystable` et `sidewaysfigure` qui permettent de placer les objet à l'italienne (en paysage), en faisant également tourner le titre.

```
\begin{sidewaystable}
   \caption{titre}
   \begin{tabular}
      […]
   \end{tabular}
\end{sidewaystable}
```

Formules mathématiques

L'utilisation des extensions de l'AMS (American Mathematical Society), et en particulier `amsmath` et `amsfonts`, est fortement recommandée.

```
\usepackage{amsmath,amssymb,amsfonts}
```

Le mode mathématique ne gère pas de manière correct les opérateurs Unicode (en particulier l'alignement et l'espacement), par exemple le caractère « × » n'est pas géré correctement, il faut utiliser les commandes (`\times` ici).

Mode mathématique

les formules mathématiques sont incluses dans des environnements : `math` pour une formule dans le texte, `displaymath` pour une formule hors alinéa, et `equation` pour une formule hors alinéa numérotée. On peut utiliser les abréviations suivantes :

- `$…$` pour une formule dans le texte ;
- `\[…\]` pour une formule hors alinéa non-numérotée.

Exemple

```
\begin{equation}
   \label{eq_gas_pft}
   PV = nRT
\end{equation}
Dans l'équation~\ref{eq_gas_pft},
dite \og équation des gaz parfaits \fg, ...
```

Fonctions

Par défaut, les lettres sont considérées comme des variables et sont mises en italique. Les fonctions sont en romain, et sont des commandes simples à retenir : une contre-oblique et l'abréviation de la fonction, par exemple `\sin`, `\cos`, `\tan`, `\cot`, `\arcsin`, `\arccos`, `\arctan`, `\coth`, `\sinh`, `\cosh`, `\tanh` `\ln`, `\log`, `\exp` `\max`, `\min`, `\sup`, `\inf`, `\lim` `\ker`, `\deg`, …

Si vous voulez définir une nouvelle fonction, il faut utiliser la commande `\DeclareMathOperator`[4] :

```
\DeclareMathOperator{\commande}{texte}
```

par exemple

```
\DeclareMathOperator{\acos}{acos}
```

Polices mathématiques

Les polices mathématiques sont différentes des polices de texte. Les instructions permettant de changer de police sont donc différentes.

Choix de la police mathématique

Type de police	Commande	Type de police	Commande
romain	`\mathrm{`*texte*`}`	supérieures (exposant)	`^{`*texte*`}`
sans empattement	`\mathsf{`*texte*`}`	inférieures (indice)	`_{`*texte*`}`
police type machine à écrire	`\mathtt{`*texte*`}`	lettres ajourées	`\mathbb{`*texte*`}`[5]
italique	`\mathit{`*texte*`}`	écriture calligraphique	`\mathcal{`*texte*`}`
gras	`\mathbf{`*texte*`}`	gothique	`\mathfrak{`*texte*`}`[5]

Les lettres grecques sont obtenue en écrivant leur nom en anglais avec une contre-oblique :

- bas-de-casse : `\alpha \beta \gamma \delta \epsilon \zeta \eta \theta \iota \kappa \lambda \mu \nu \xi \pi \rho \sigma \tau \upsilon \phi \chi \psi \omega` ;
- capitales : `\Gamma \Delta \Lambda \Xi \Pi \Sigma \Upsilon \Phi \Psi \Omega`.

Certaines lettres sont identiques aux lettres latines et ne sont donc pas définies, comme le omicron, qui est en fait identique au « o », et de nombreuses capitales (l'alpha capitale est identique au A etc.).

Certaines lettres ont des variantes.

Variantes des lettres grecques

Première variante	Caractère	Seconde variante	Caractère
`\epsilon`	ϵ	`\varepsilon`	ε
`\theta`	θ	`\vartheta`	ϑ
`\pi`	π	`\varpi`	ϖ
`\rho`	ρ	`\varrho`	ϱ
`\sigma`	σ	`\varsigma`	ς
`\phi`	ϕ	`\varphi`	φ

Le mode mathématique gère différemment les espaces et les accents. Lorsque l'on veut mettre du texte normal en mode mathématique (par exemple entre deux expressions), on utilise la commande `\text{…}`[4].

Opérateurs, quantificateurs, relations et symboles

Opérateurs arithmétiques

Instruction	Symbole	Instruction	Symbole
`+`	$+$	`\div`	$\div$
`-`	$-$	`:`	$:$
`\times`	$\times$	`\Im`, `\Re`	$\Im$, $\Re$
`\cdot`	$\cdot$	`\otimes`	$\otimes$
`/`	$/$	`\wedge`	$\wedge$

Grands opérateurs arithmétiques

Instruction	Symbole	Instruction	Symbole
`\frac{a}{b}`	$\frac{a}{b}$	`\prod_{i=1}^{n} u_i`	$\prod_{i=1}^{n} u_i$
`\sqrt{a}`	$\sqrt{a}$	`\int_{a}^{b} f(x)dx`	$\int_{a}^{b} f(x)dx$
`\sqrt[n]{a}`	$\sqrt[n]{a}$	`\oint_{\Lambda} f(x)dx`	$\oint_{\Lambda} f(x)dx$
`\sum_{i=1}^{n} u_i`	$\sum_{i=1}^{n} u_i$		

L'extension `amsmath` fournit les commandes `\iint` et `\iiint` pour les intégrales doubles et triples.

Dérivation

Instruction	Symbole
`\nabla`	∇
`\partial`	∂

Certains quantificateurs et relations ont leur négation, par exemple `\exists` et `\nexists`. S'il n'existe pas, on peut faire précéder la commande par `\not` (par exemple `\not\exists`).

Quantificateurs

Instruction	Symbole
`\forall`	$\forall$
`\exists`	$\exists$
`\nexists`	$\nexists$

Relations

Instruction	Symbole	Instruction	Symbole
`=`	$=$	`\neq`	$\neq$
`\equiv`	$\equiv$	`\ll`, `\gg`	$\ll$, $\gg$
`\approx`	$\approx$	`\lll`, `\ggg`[6]	$\lll$, $\ggg$
`\simeq`	$\simeq$	`\Rightarrow`, `\Longrightarrow`	$\Rightarrow$, $\Longrightarrow$
`\propto`	$\propto$	`\iff`	$\iff$
`\leq`, `\geq`	$\leq$, $\geq$	`\leqslant`, `\geqslant`[6]	$\leqslant$, $\geqslant$
`\pm`, `\mp`	$\pm$, $\mp$	`\to`	$\to$

Ensembles

Instruction	Symbole	Instruction	Symbole
`\varnothing`	$\varnothing$	`\subset`	$\subset$
`\cap`	$\cap$	`\subseteq`	$\subseteq$
`\cup`	$\cup$	`\in`	$\in$

Opérateurs logiques

Instruction	Symbole
`\lnot`	$\lnot$
`\land`	$\land$
`\lor`	$\lor$

Géométrie

Instruction	Symbole	Instruction	Symbole
`\wedge`	$\wedge$	`\sphericalangle`	$\sphericalangle$
`\angle`	$\angle$	`\perp`	$\perp$
`\measuredangle`	$\measuredangle$		

Flèches

Instruction	Symbole	Instruction	Symbole
`\leftarrow`, `\longleftarrow`	$\leftarrow$, $\longleftarrow$	`\xrightarrow[a]{b}`, `\xleftarrow[a]{b}`[4]	$\xrightarrow[a]{b}$, $\xleftarrow[a]{b}$
`\rightarrow`, `\longrightarrow`	$\rightarrow$, $\longrightarrow$	`\leftrightarrows`, `\leftrightharpoons`	$\leftrightarrows$, $\leftrightharpoons$
`\leftrightarrow`, `\longleftrightarrow`	$\leftrightarrow$, $\longleftrightarrow$	`\uparrow`, `\downarrow`, `\updownarrow`	$\uparrow$, $\downarrow$, $\updownarrow$
`\nwarrow`, `\nearrow`, `\searrow`, `\swarrow`	$\nwarrow$, $\nearrow$, $\searrow$, $\swarrow$	`\Uparrow`, `\Downarrow`, `\Updownarrow`	$\Uparrow$, $\Downarrow$, $\Updownarrow$

On fera attention à la présence éventuelle d'un *s*, ainsi qu'à l'ordre des mots *left* et *right*. Si l'on met une capitale à l'instruction de la flèche, on obtient en général une flèche double.

Divers

Instruction	Symbole	Instruction	Symbole
`\infty`	∞	`\bullet`	$\bullet$
`\circ`	$\circ$	`\cdots`, `\ldots`	$\cdots$, $\ldots$

Délimiteurs

Les délimiteurs classiques sont `(…)`, `[…]`, `\{…\}`, `|…|`, `\|…\|` (pour ||…||), `\langle…\rangle` (pour <…>).

Pour avoir des délimiteurs adaptables en taille à leur contenu, on met `\left` devant le délimiteur d'ouverture et `\right` devant le délimiteur de fermeture, par exemple `\left ( \frac{a}{b} \right )`.

Si l'on ne veut qu'un délimiteur d'ouverture (par exemple une grande accolade), on termine l'expression par `\right .` (le point n'est pas affiché).

Accents mathématiques

Accents mathématiques

Instruction	Résultat	Instruction	Résultat
`\acute{a}`	$\acute{a}$	`\tilde{a}`	$\tilde{a}$
`a', a^\prime`	a', a'	`\dot{a}, \ddot{a}`	$\dot{a}, \ddot{a}$
`\hat{a}`	$\hat{a}$	`\widehat{ABC}`	$\widehat{ABC}$
`\bar{a}`	$\bar{a}$	`\overline{AB}`	$\overline{AB}$
`\grave{a}`	$\grave{a}$	`\underline{AB}`	$\underline{AB}$
`\overbrace{1,\ldots,n}^{a}`	$\overbrace{1,\ldots,n}^{a}$	`\underbrace{1,\ldots,n}_{b}`	$\underbrace{1,\ldots,n}_{b}$
`\vec{a}`	$\vec{a}$	`\overrightarrow{AB}`	$\overrightarrow{AB}$

Il existe des accents spécifiques aux mathématiques et à la physique.

Les instructions `\imath` et `\jmath` donnent un *i* et un *j* sans point, ce qui évite la superposition d'un accent au point.

Mise en forme des formules

Le mode mathématique supprime tous les espaces. Il faut donc les forcer.

Espaces typographiques en mode mathématiques

Instruction	Espace	Instruction	Espace
`\?`	normale (justifiante)	`\;`	grande
`\,`	fine	`\!`	négative (permet le rapprochement d'objets)

L'extension `amsmath` permet d'empiler des éléments.

Placement vertical

Instruction	Résultat
`\overset{a}{X}`	$\overset{a}{X}$
`\underset{b}{X}`	$\underset{b}{X}$
`\overset{a}{\underset{b}{X}}`	$\overset{a}{\underset{b}{X}}$

L'environnement `array` est l'équivalent de `tabular` pour le mode mathématiques.

Matrices

L'extension `amsmath` fournit les environnements suivants :

- `matrix` : matrice sans délimiteur ;
- `pmatrix` : matrice entre parenthèses (…) ;
- `vmatrix` : matrice entre barres |…| ;
- `Vmatrix` : matrice entre doubles barres ‖…‖ ;
- `bmatrix` : matrice entre crochets […] ;
- `Bmatrix` : matrice entre accolades {…}.

La ligne d'une matrice se termine par un `\\` (double contre-oblique) et sur une ligne, les éléments sont séparés par une esperluette `&`

```
\begin{pmatrix}
   a_1 & b_1 \\
   a_2 & b_2
\end{pmatrix}
```

Pour les ellipses, on dispose des symboles suivants.

Ellipses pour matrices

Instruction	Symbole	Instruction	Symbole
`\cdots`	$\cdots$	`\ldots`	$\ldots$
`\vdots`	$\vdots$	`\ddots`	$\ddots$

Physique

En mode mathématiques, la commande `\mho` affiche le symbole du mho, $\mho$, et `\hbar` affiche $\hbar$.

L'extension `[frenchb]{babel}` fournit la commande `\nombre{`*nombre*`}` qui met en forme les nombres longs.

L'extension `SIunits` permet la mise en forme les unités :

```
\usepackage[cdot]{SIunits}
…
\unit{nombre}{unité}
```

Les unités sont composées à partir des unités SI (`\meter`, `\kilogram`, `\second`, `\ampere`, `\kelvin`, `\mol`, `\candela`) et des suffixes (`\nano`, `\micro`, `\kilo`, `\mega`, `\giga`, …). Certaines unités non SI ou dérivées du SI sont disponibles (`\gram`, `\liter`, `\hertz`, `\newton`, `\pascal`, `\ohm`, `\celsius`, `\angstrom`, `\rad`, `\degree`, `\arcminute`, `\arcsecond`, …).

On peut composer les unités. Si elles sont accolées ou séparées par des espaces, les symboles sont accolés (multiplication). Si elles sont séparées par un point « . », le séparateur dépend de l'option d'appel de l'extension (espace par défaut, point multiplicateur avec l'option `cdot`). On utilise `\per` pour la division.

On peut élever à une puissance quelconque avec `\power{`*unité*`}{`*puissance*`}`. On peut aussi utiliser `\reciprocal` avant l'unité pour élever à la puissance -1, `\squared` ou `\cubed` après l'unité pour les puissances 2 et 3, `\rpsquared` ou `\rpcubed` après l'unité pour les puissances -2 et -3.

Certaines unités composées sont déjà définies (`\squaremeter`, `\kilowatthour`, …).

On peut aussi utiliser les unités hors de la commande `\unit`.

Exemple

```
\unit{\nombre{1000}}{\meter} = \unit{1}{\kilo\meter}

\unit{1}{\newton} = \unit{1}{\kilogram \meter \second\rpsquared}
   = \unit{1}{\kilogram.\meter \per \second\squared}

$\unit{5\cdot 10^3}{\newton}$

La vitesse s'exprime en mètres par seconde (\meter\per\second)
```

L'extension `sistyle` est plus simple d'utilisation : elle définit une commande similaire

```
\SI{nombre}{unité}
```

mais unité est écrite comme une formule mathématique

```
\SI{1}{N}=\SI{1}{kg.m/s^2}
```

Si l'on veut utiliser le point décimal dans l'unité, il faut employer `\pnt`. On dispose également de quelques unités et suffixes particuliers (`\angstrom`, `\micro`, `\degC`, `\arcdeg`, `\arcmin`, `\arcsec`).

Formule chimique

L'extension `mhchem` permet d'écrire simplement des formules compactes et semi-développées, ainsi que des équations de réaction.

```
\usepackage[version=3]{mhchem}
...
\ce{1/2H2O}
\ce{^{227}_{90}Th+}
```

donnent respectivement $\frac{1}{2}\mathrm{H_2O}$ et $^{227}_{90}\mathrm{Th}^{+}$. Les nombres après les symboles chimiques sont en indice, les + et - sont en exposant, et l'on peut utiliser la notation mathématique `^{…}` et `_{…}` pour forcer la position. Le `\ce` est un mode mathématiques, on peut par exemple écrire `\ce{\frac{n}{2}H2O}`. Par ailleurs :

- la liaison simple se marque avec un `-` ou `\sbond`, la double avec un `=` ou `\dbond`, et la triple avec un `#` ou `\tbond` ;
- la flèche de réaction se marque `->` ;
  - on peut mettre du texte au dessus avec `->[`*`texte au dessus`*`]`,
  - du texte au dessus et en dessous avec `->[`*`texte au dessus`*`][`*`texte en dessous`*`]` ;
- la double flèche d'équilibre se note `<=>` ;
- un chapeau `^` entouré de deux espaces indique un dégagement de gaz ? ; un `v` entouré de deux espaces indique une précipitation ? ;
- un astérisque « * » ou un point « . » est transformé en un point centré « · ».

L'extension *ppchtex* permet également d'écrire des formules et équations :

```
\usepackage{etex}
\usepackage{m-pictex,m-ch-en}
...
\chemical{2H_2,+,O_2,->,2H_2O}
```

La syntaxe générale est `\chemical{`*`élément 1,élément 2,…`*`}` où *élément n* est :

- une molécule ou une partie de molécule : on utilise ici la notation mathématique ;
- une liaison simple `-`, double `--` ou triple `---` ;
- une flèche de réaction : `->` (réaction simple), `<=>` (équilibre).

Si la formule est dans un mode mathématique hors paragraphe (`\[...\]` ou environnement `equation`), on peut mettre du texte au-dessus ou en dessous.

```
\chemical{texte principal}{texte dessous}
\chemical{texte principal}{texte dessus}{texte dessous}
```

Code source

L'extension `listings` permet de mettre du code source. On ne peut pas utiliser de caractères Unicode dans le code mise en forme par les commandes de cette extension.

Le code source est placé dans un environnement `lstlisting` ; la mise en forme stricte (y compris les espaces et les retours de ligne) est respectée, et les commandes LaTeX ne sont pas interprétées.

On définit le langage ainsi qu'un caractère d'échappement juste après l'appel de l'extension :

```
\usepackage{listings}
\lstset{language=TeX,
   basicstyle=\ttfamily\small,
   columns=flexible,
   escapechar=+}
```

Dans l'exemple ci-dessus, on indique :

- que l'on écrit du TeX ou du LaTeX, ce qui permettra à l'extension de reconnaître les mots-clefs et d'appliquer une mise en forme spécifique ;
- que le code sera en police Teletype de corps plus petit que le texte ;
- que les colonnes sont « flexibles », c'est-à-dire que l'écriture peut être un plus compacte au détriment éventuellement de l'alignement vertical ;
- que le caractère d'échappement est « + ».

Le texte compris entre deux caractères d'échappement est interprété par LaTeX, par exemple dans

```
\begin{lstlisting}
{\large +\emph{Texte en corps plus grand}+}
\end{lstlisting}
```

la séquence `{\large` et le `}` final seront imprimés tels quels, tandis que le `\emph{Texte en corps plus grand}` sera interprété (on aura donc le contenu du bloc en italique). Il faut évidemment choisir un caractère d'échappement qui ne sera pas dans le code.

On peut mettre du code au sein d'un texte, avec la commande `\lstinline`*`c...c`*. Le *c* indique un caractère qui marque le début et la fin du code ; il peut être choisi arbitrairement, mais ne doit évidemment pas faire partie du code. Par exemple

```
Pour mettre du texte en emphase,
on utilise la commande \lstinline-\emph-.
```

Un grand nombre de langages sont disponibles : `Pascal`, `Fortran`, `Basic`, `C`, `C++`, `Ada`, `Scilab`, `HTML`, `Java`, … On peut indiquer des variantes, par exemple

```
\lstset{language=[95]Ada}
```

```
\lstset{language=[Visual]C++}
\lstset{language=[77]Fortran}
```

On peut aussi indiquer les options lors de l'appel de l'environnement `lstlisting` :

```
\begin{lstlisting}[language=XML,escapechar=?]
...
\end{lstlisting}
```

Notes

1. le mot espace, lorsqu'il désigne un écart horizontal entre deux mots dans le document final, est au féminin ; une espace était une pièce destinée à écarter les caractère de plomb
2. *ragged right* pour « drapeau droit » (littéralement « en lambeau à droite »)
3. l'alignement vertical est défini par rapport à la ligne de base de la cellule de référence ; on peut placer un tableau dans une cellule, afin que les autres cellules les considère comme une ligne unique
4. cette commande nécessite l'extension `amsmath`
5. cette commande nécessite l'extension `amsfonts`
6. cette commande nécessite l'extension `amssymb`

Voir aussi

- Version PDF du *Vade mecum*

Conversion vers d'autres formats

À proprement parler, un fichier source en Latex peut être converti directement vers deux formats :

- le format DVI en utilisant la commande *latex*, le format d'origine,
- le format PDF en utilisant la commande *pdflatex*, plus récent.

En utilisant d'autres logiciels libres disponibles sur internet, vous pouvez facilement convertir les formats DVI et PDF en d'autres formats de document. En particulier, vous pouvez convertir vers le format PostScript utilisé par les imprimantes au moyen de *dvips* qui fait partie de votre distribution Latex.

Certains environnements de composition de documents en Latex (comme LyX) vous permettrons de générer directement un fichier au format PostScript. Cependant, ils génèrent de façon interne un fichier intermédiaire au format DVI, en suivant le cheminement LaTeX --> DVI --> PostScript.

Il est également possible de créer un fichier PDF à partir d'un fichier DVI et vice-versa.

Il ne semble pas logique de créer un fichier en deux étapes quand vous pouvez le créer immédiatement, mais certains utilisateurs pourraient avoir besoin de cette étape intermédiaire, comme nous l'avions évoqué dans les premiers chapitres, le format que vous pouvez produire dépend des formats des images vous avez incluses (EPS pour DVI, PNG et JPG pour PDF). Dans ce qui suit, nous allons discuter des différents formats de fichier en vous expliquant comment obtenir une conversion du fichier en Latex vers ces différents formats.

D'autres formats peuvent être obtenus, tel que RTF (utilisé dans l'univers de Microsoft) et HTML. Cependant, ces documents sont générés à partir d'un logiciel qui analyse et interprète les fichier Latex, et qui ne met pas en œuvre toutes les fonctionnalités disponibles pour les formats DVI et PDF. N'importe comment, ils fonctionnent, et peuvent être des outils d'une importance cruciale pour la correspondance avec d'autres personnes qui ne savent pas lire un fichier en Latex.

Portable Document File ou PDF

Conversion du DVI en PDF

```
dvipdfm fichier.dvi
```

créera `fichier.pdf`. Un autre moyen est de passer par la production d'un fichier PS:

```
dvi2ps fichier.dvi
ps2pdf fichier.ps
```

Vous obtiendrez un fichier nommé *fichier.ps* que vous pourrez détruire.

Fusion de PDF

Si vous avez créé plusieurs documents au format PDF et vous désirez les fusionner en un seul fichier PDF, vous pouvez, à condition que Ghostscript soit installé, taper la ligne de commande suivante:

sous Windows:

```
gswin32 -dNOPAUSE -sDEVICE=pdfwrite -sOUTPUTFILE=Merged.pdf -dBATCH 1.pdf 2.pdf 3.pdf
```

sous Linux:

```
gs -dNOPAUSE -sDEVICE=pdfwrite -sOUTPUTFILE=Merged.pdf -dBATCH 1.pdf 2.pdf 3.pdf
```

PostScript

à partir du PDF

```
pdf2ps fichier.pdf
```

à partir du DVI

```
dvi2ps fichier.dvi
```

Rich Text Format ou RTF

Un fichier LaTeX peut être converti en un fichier RTF. Ce dernier peut être ouvert par Word de Microsoft et OpenOffice.org Writer. Cette conversion est réalisée au moyen de latex2rtf (http://latex2rtf.sourceforge.net/) [archive], qui peut fonctionner sur n'importe quelle plateforme informatique.

Le programme lit le fichier source en Latex, et traduit vers le format RTF en imitant le comportement du programme LaTeX . Le logiciel `latex2rtf` supporte la plupart des possibilités courantes de Latex, telles que la mise en page, certaines commandes mathématiques, les importations d'images au formats EPS, PNG ou JPG, et la réalisation de tableaux. Par ailleurs, le programme n'autorise qu'un nombre limité d'extensions (ou de paquets), tels que *varioref* et *natbib*, et beaucoup d'autres paquets ne sont pas supportés.

Le programme latex2rtf est simple d'emploi. La version sous Windows dispose d'une interface graphique très agréable. La version en ligne de commande est gratuite pour toutes les plateformes, et peut être employée sur un fichier *fichier.tex* en tapant:

```
latex fichier
bibtex fichier # si vous utilisez bibtex
latex2rtf fichier
```

Les deux commandes `latex` et (si nécessaire) `bibtex` doivent être exécutées ***avant*** `latex2rtf`, parce que les fichiers `.aux` et `.bbl` sont nécessaires pour produire correctement le fichier RTF. Cette conversion créera le fichier `fichier.rtf`, que vous pourrez ouvrir à l'aide de nombreux traitements de texte modernes. Les utilisateurs de Microsoft Word peuvent sauver ce fichier sous fichier.doc pour pouvoir utiliser pleinement toutes les fonctionnalités du logiciel, telle que *suivi des changements* (Track Changes).

Conversion vers des formats d'image

Portable Network Graphics ou PNG

Il existe plusieurs méthodes permettant de convertir un fichier Latex en un fichier au format PNG.

- Vous pouvez le convertir en un fichier PDF, puis ouvrir le fichier obtenu avec GIMP. Le logiciel vous demandera quelle page vous voulez convertir, et si vous désirez employer l'anticrénelage, choisissez un *lissage fort*, pour obtenir quelque chose de semblable au fichier PDF lorsqu'il est affiché à l'écran.

Essayez différentes résolutions afin d'adapter à vos besoins, mais le 100 dpi devrait amplement suffire. Une fois l'image importée dans GIMP, vous pouvez si vous le désirez, effectuer quelques traitements graphiques, puis la sauvegarder sous l'un des formats d'image supportés par GIMP, comme PNG par exemple.

- Une autre méthode consiste à utiliser *dvipng* (http://savannah.nongnu.org/projects/dvipng/) [archive] qui fonctionne de façon identique à *dvipdfm*. Vous pouvez réduire la taille du fichier obtenu en employant le logiciel optipng (http://optipng.sourceforge.net/) [archive].

Scalable Vector Graphics ou SVG

Sous unix, on convertit dans un premier temps le fichier Latex en PostScript avec *dvips*, puis on lui applique un programme bash Ps2svg.sh dont les instructions sont les suivantes:

```
#!/bin/bash

# Vous avez besoin de gs-common, pstoedit et skencil,
# pour que ce programme fonctionne
export BASENAME="`basename $1 .ps`";

# Outline fonts
eps2eps -dNOCACHE ${BASENAME}.ps ${BASENAME}2.ps

# détermine la taille de l'image
ps2epsi  ${BASENAME}2.ps  ${BASENAME}.ps
rm ${BASENAME}2.ps

# convertit vers le format de Sketch
pstoedit -f sk  ${BASENAME}.ps  ${BASENAME}.sk

# convertit vers SVG
skconvert  ${BASENAME}.sk  ${BASENAME}.svg
```

On peut également employer *dvisvgm*[24] (http://dvisvgm.sourceforge.net/), un utilitaire en source disponible qui convertit le dvi en svg.

Conversion vers des formats hypertextes

Conversion vers le HTML

Le langage HTML, ne pouvant qu'afficher au départ que du texte et des images, la conversion d'un fichier en Latex vers ce format n'est pas simple. Il est souvent difficile voire même impossible de faire respecter les dactylographiques, ou la mise en page d'un texte. Ne parlons pas du rendu des

formules mathématiques ou même de leur alignement avec le texte. Les formules mathématiques sont ou bien représentées par des caractères d'une police donnée ou des images.

Plusieurs logiciels permettent une telle conversion:

- *TeX4ht* TeX4ht (http://www.cse.ohio-state.edu/~gurari/TeX4ht/mn.html) [archive]
- *Hevea* hevea (http://pauillac.inria.fr/~maranget/hevea/papers/hevea/) [archive]
- *Latex2html* Latex2html (http://www.latex2html.org/) [archive] qui n'est pas un logiciel libre,
- *ttwp* ttwp (http://www.xm1math.net/ttwp/index.html) [archive]
- Pandoc pandoc (http://johnmacfarlane.net/pandoc/) [archive]

Conversion vers le MathML

Le langage à balises MathML basé sur XML a été conçu pour pouvoir représenter les symboles mathématiques et transmettre des textes mathématiques par internet. Pour l'instant, peu de navigateurs supportent toutes les fonctionnalités de MathML, mais il existe des logiciels permettant de convertir un fichier Latex en MathML :

- *TeX4ht* TeX4ht (http://www.cse.ohio-state.edu/~gurari/TeX4ht/mn.html) [archive]
- *Hevea* hevea (http://pauillac.inria.fr/~maranget/hevea/papers/hevea/) [archive]

Glossaire de typographie

Sommaire : Haut - A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Voir aussi

Conventions typographiques de Wikipédia

A

Alinéa

L'alinéa désigne trois choses :

1. Le retour à la ligne *(a linea)*, qui est suivi en typographie d'un retrait de paragraphe, dans la composition dite « en alinéa » ;
2. Le texte situé entre deux retours à la ligne ; dans ce sens, il y a souvent une confusion avec le *paragraphe* (voir plus bas) ;
3. Le renfoncement du début de la première ligne proprement dit ; on utilise parfois les termes « indentation » ou « retrait de paragraphe ».

Les autres découpages du texte sont le paragraphe et la phrase, avec l'ordre hiérarchique : paragraphe > alinéa > phrase. Un paragraphe contient plusieurs alinéa (plusieurs retours à la ligne avec renfoncement) et un alinéa plusieurs phrases, mais dans la pratique, il est rare d'avoir besoin des trois niveaux de séparation de texte.

Lorsqu'il n'y a pas de renfoncement (composition en pavé ou en drapeau), on distingue les alinéa par un interligne plus grand (il n'est alors plus possible de distinguer alinéa et paragraphe).

En composition typographique, l'alinéa (renfoncement) vaut un cadratin à un cadratin et demi.

Approche

L'approche est l'espacement entre les lettres d'un mot, qui évitent qu'elles soient collées ; elle est proportionnelle au corps. Si l'outil informatique permet de modifier cette approche, cette solution est à éviter car elle rend la lecture difficile.

Les lettres présentant des diagonales ont un crénage permettant de les rappocher, comme par exemple « AV » (le haut du V est à l'aplomb du bas du A).

Voir aussi chasse, hauteur d'x.

B

B (écriture de type ~)

L'écriture de type B est une écriture normalisée utilisée en dessin technique (norme ISO 3098). Le *cadratin*, appelé « hauteur nominale » *h*, est indiqué en millimètres (de 2,5 à 20 mm) : une « écriture B, droite de 10 » désigne des lettres romaines de 10 mm de haut. On utilise des caractères linéaux (sans empattement, sans plein ni délié, de type Arial).

La *hauteur d'x c* vaut 0,7 fois le cadratin ($c = 0{,}7 \cdot h$), l'épaisseur du trait *d* vaut un dixième du cadratin ($d = 0{,}1 \cdot h$). La chasse (largeur) des capitales va de $0{,}4 \cdot h$ pour le J à *h* pour la W (sauf le I qui a bien sûr une largeur *d*), et vaut $0{,}6 \cdot h$ pour la plupart des caractères. La largeur des chiffres vaut un demi-cadratin sauf pour le 1.

L'approche *a* vaut deux fois l'épaisseur du trait ($a = 2 \cdot d$), sauf lorsqu'il y a deux traits verticaux adjacent auquel cas il veut $2{,}5 \cdot d$, et en cas « d'emboîtement » où l'approche est nulle ($a = 0$, cf. *crénage*). L'interligne *b* vaut au moins $1{,}4 \cdot h$. L'espacement entre les mots vaut un cadratin *h*.

Bas-de-casse

Désigne les caractères placés en bas de la *casse*, c'est-à-dire les lettres minuscules.

Certaines police ont des chiffres bas-de-casse, également appelés chiffres elzéviriens : 0123456789 (pour un bon rendu, la police Hoefler Text ou Georgia doivent être installées).

Voir aussi *capitale*.

Belle page

Sur un document en recto-verso, la belle page désigne une page impaire, le recto du feuillet. Le *faux titre*, le *titre*, la page indiquant les parties, et parfois la première page des chapitres sont en belle page.

Voir aussi *fausse page*.

Blanc

marge.

Bourdon

Oubli d'un ou plusieurs mots, voire d'une phrase ou d'un paragraphe.

Voir aussi *coquille*, *doublon*.

C

Cadratin

Longueur égale au *corps* du texte ; largeur de la lettre M. Cette unité est appellée `em` en CSS et dans LaTeX.

Espace cadratin, tiret cadratin (—).

Capitale

Les capitales sont les lettres placées en tête de la *casse* (du latin *caput, capitis*, la tête), c'est-à-dire les grandes lettres. On distingue :
- LES GRANDES CAPITALES ;
- LES PETITES CAPITALES.

Les capitales sont accentuées (ÀÂÄÉÈÊËÎÏÔÖÙÛÜŸ) et ligaturées (ÆŒ).
Voir aussi *majuscule* et *bas-de-casse*.

Caractère mobile ou typographique

En typographie, un caractère est une pièce métallique qui va servir à l'impression. La pièce présente un relief, ou « œil » (pluriel : œils et non pas yeux), sur lequel va se déposer l'encre. Les caractères sont placés dans une *casse* ; lorsque le typographe compose un texte, il prend les caractères de la casse, il les place dans un *composteur* pour former la ligne. Puis, la ligne est transférée dans un châssis pour l'impression ; la page prête à être imprimée (châssis et caractères mobiles) est appelée « forme ».
Ce procédé a été rendu obsolète par la photocomposition et la pagination assistée par ordinateur. Les premiers caractères mobiles ont été utilisés en Chine au XII^{e} siècle ; en Europe, ils ont été découverts par Gutenberg au XV^{e} siècle.
Certains caractères n'ont pas d'œil et servent uniquement à régler l'espacement entre les mots : ce sont les *espaces*, mot féminin.
Par extension, un « caractère » désigne maintenant un signe imprimable ou affichable : lettre, signe de ponctuation, espace, …

Casse

Boîte dans laquelle on dispose les *caractères* mobiles. En tête de casse se trouvent les grandes lettres, dites *capitales* (du latin *caput, capitis*, la tête, souvent appelées à tort « majuscules »), et en bas de la casse les… *bas de casse* (minuscules).

Césure

Lorsque le texte est *composé* de manière justifiée, il faut parfois couper les mots pour respecter à la fois le *gris optique* et la *justification* ; c'est la césure. La césure se marque pour un signe « - » appelé *division*, placé en fin de ligne. Elle suit des règles très précises, notamment :
- la coupure ne doit pas changer la prononciation ;
- les mots composés sont coupés selon l'étymologie : trans-action, cis-alpin, in-jonction, atmo-sphère, caou-tchouc ; s'il y a un trait d'union, ils sont coupés au trait d'union ;
- les mots simples sont coupés entre les syllabes : ac-tion, al-pin, jonc-tion ;
- on ne coupe pas les noms propres ;
- on ne laisse pas deux lettre seules en fin de ligne, ni une lettre seule en début de ligne, et on évite d'avoir des césures sur plus de trois lignes consécutives ;
- con-cu-vi : par bienséance, aucune ligne comportant une césure ne doit se terminer par une de ces trois syllabes.

Lorsque l'on compose par ordinateur (traitement de texte ou logiciel de PAO), la césure est soit automatique, soit l'on indique l'endroit possible de la césure avec une division conditionnelle (Ctrl+- sous Windows ou ⌘+- sous MacOS), mais on ne place jamais la division à la main.

Chasse

La chasse désigne la largeur d'un caractère. Elle est proportionnelle au *corps*, mais varie d'une *fonte* à l'autre ; comparer :
- Chasse (Arial corps 10 pt) ;
- Chasse (Arial corps 12 pt) ;
- Chasse (Times corps 12 pt) ;
- Chasse (Courier corps 12 pt).

Dans les polices à chasse variable (les plus courantes), la lettre minuscule la plus large est le « m » et la plus étroite est le « i ». Les fontes de type machine à écrire (comme Courier) ont au contraire une chasse fixe.
Voir aussi *approche*, *hauteur d'x*.

Cicéro, ou douze

Longueur valant douze points didot, soit 4,51 mm.

Colophon

Le colophon est une note située à la fin d'un ouvrage de luxe, en *belle page*, et indiquant ses principales caractéristiques (qualité du papier, nombre d'exemplaires, reliure, ex-folio, …).
Voir *parties d'un ouvrage*.

Composition

La composition est la manière d'arranger un texte ; le texte peut être :
- composition en *alinéa* : les caractères occupent toute la largeur de la ligne *(justification)*, sauf la première ligne d'un paragraphe qui présente un renfoncement, et la dernière ligne qui est dite « creuse » (elle n'occupe pas toute la justification, elle est alignée à gauche) ; cela nécessite des *espaces* variables (espaces dites « justifiantes ») et des *césures* ;
- composition en pavé : comme précédemment, mais la première ligne est justifiée ;
- composé en drapeau droit : les caractères sont alignés au *fer* à gauche, les espaces sont fixes, il n'y a pas de césure, les lignes ne se terminent pas systématiquement au bord droit ;
- composé en drapeau gauche : les caractères sont alignés au fer à droite, les espaces sont fixes, il n'y a pas de césure, les lignes ne commencent pas systématiquement au bord gauche ;
- centré : les milieux des lignes sont à la même position.

Composteur

Un composteur est une sorte de règle permettant de bien placer les caractères pour bien former une ligne ; la largeur de la ligne, ou *justification*, est déterminée par la position des bords mobiles, ou *fers*.

Coquille

Une coquille est une faute dans la composition typographique, consistant à inverser des lettres ou à se tromper de lettre. Avec la fin de la

composition à *caractère mobile*, la coquille a été remplacée par la faute de frappe, ou la reconnaissance optique erronée.
Voir aussi *bourdon*, *doublon*.

Corps de texte
Hauteur totale de toutes les lettres, hampe et jambage compris. Elle est en général exprimée en *points*.
Corps de 10 points, Corps de 12 points.
Voir aussi *approche*, *cadratin*, *chasse*, *hauteur d'x*.

Crénage
Voir *approche*.

D

Demi-cadratin
longueur à la moitié du *cadratin* ; largeur de la lettre N, parfois abrégée « en » en anglais.
Espace demi-cadratin, *tiret* demi-cadratin (–)

Didot
Famille de typographes et imprimeurs : François Didot (1689–1757), François-Ambroise Didot (1730–1804), inventeur du *point* didot, et Firmin Didot (1764–1836), inventeur des polices Didot.

Division
La division est un *caractère* : c'est le signe « - » marquant la *césure*. Il se distingue du *trait d'union* qui lui est présent même en milieu de ligne. Le terme division désigne parfois par extension la césure. La division est plus courte que le *tiret*.

Doublon
Mot ou passage composé deux fois. Ce type d'erreur tend à se multiplier avec l'outil informatique et son copier-coller.
Voir aussi *bourdon*, *coquille*.

Douze
Voir *cicéro*.

Drapeau
Voir *composition*.

E

Elzévir
voir *famille* et *bas-de-casse*.

Em
voir *cadratin*.

Empagement
largeur des blancs (marges).

Ex
voir *hauteur d'x*.

F

Famille
Le terme « famille » est utilisé dans deux sens différents :
- en typographie classique, la famille de *caractères* est une classification des *polices* :
  - famille des elzévirs (ou garaldes), à empattements triangulaires, avec des pleins et des déliés d'épaisseur modérée (Times, Garamond),
  - famille des *didots* (ou didones), à empattement filliformes et des pleins et déliés marqués (Bodoni, Didot),
  - famille égyptienne (ou mécanes) : à empattement rectangulaires, absence de pleins et de déliés (Rockwell, Clarendon, polices type `machine à écrire`),
  - famille des antiques (ou linéales) : sans empattement, absence de pleins et déliés (Arial, Helvetica, Univers),
  - les scriptes : imitent l'écriture manuscrite ;
- de nos jours, avec la confusion entre les termes *police* et *fonte*, la famille de fonte désigne la police : « Times corps 12 romain » est une fonte, sa « police », ou « famille de fonte », est Times.

Fausse page
Dans les ouvrages en recto-verso, le terme de fausse page désigne les pages paires, les verso des feuillets.
Voir aussi *belle page*.

Faux titre
Le faux titre est sur recto *(belle page)* du deuxième feuillet suivant la couverture (dans le cas des livres reliés, le premier feuillet est collé sur la couverture, le faux titre est donc sur le deuxième feuillet du cahier). Elle comporte uniquement le titre succint de l'ouvrage.

Voir aussi *parties d'un ouvrage*, *titre*.

Fer

Les fers sont les parties mobiles du *composteur* déterminant la largeur de la ligne, ou *justification*. Voir *Composition*.

Forme

Voir *caractère*

Fonte

En typographie, la fonte est l'ensemble des caractères d'une *police* pour un corps donné, dans une famille (*italique* ou *romain*) et une graisse (gras ou moyen) donnés. Une fonte comportait typiquement 100 000 *caractères mobiles* (9 000 « e », 300 « k », 850 « E », 75 « K », 400 « 1 », 1 800 «) », …)[1]. Les caractères de la fonte sont mis dans une boîte appelée *casse*. La fonte comprend les grandes lettres (grandes et petites *capitales*), les minuscules *(bas de casse)*, les lettres *ligaturées* et accentuées. Le nom « fonte » provient du procédé d'élaboration des caractères : on faisait couler du métal fondus dans des moules.
Exemples de fontes : *Garamond corps 10 italique*, Times corps 12 romain, **Arial corps 10 gras**, …
En langage moderne, police est synonyme de fonte.

G

Gris typographique ou optique

Le gris typographique est l'impression générale que laisse la feuille lorsqu'on la regarde en plissant les yeux. Un texte agréable à lire a un gris uniforme. Lors de la *composition*, on modère les espaces en ayant recours à la *césure*, voire exceptionnellement en rompant la *justification* : une ligne peut être légèrement plus longue ou plus courte que les autres.
Voir aussi *lézarde*.

H

Hauteur d'« x »

Souvent appelé à tort « hauteur d'œil », c'est la hauteur d'une *bas de casse* sans hampe ni jambe, c'est-à-dire la hauteur d'un « x ». Cette unité est appelée ex en CSS et dans LaTeX
Voir aussi *approche*, *chasse*, *corps*.

J

Justification

La justification est la largeur de la ligne d'écriture. Le terme « justifié » est maintenant utilisé pour indiquer que les lignes occupent toute la justification, sauf la dernière ligne d'un paragraphe qui est dite « creuse », et la première ligne qui peut présenter un renfoncement *(alinéa)* ; en typographie, on parle dans ce cas de *composition* en alinéa ou en pavé, par opposition aux compositions en drapeau ou centrées (pour lesquelles les lignes n'occupent pas toute la justification).

L

Lettrine

Une lettrine est la première lettre du premier paragraphe d'un chapitre ou d'un article de journal. Elle est composée d'un corps plus grand que le reste du texte, en général le double ou le triple. Le Haut de la lettrine est aligné avec le haut de la première ligne de texte. Dans les livres, le mot qui suit, ou parfois le groupe de mots lorsque le premier mot est court ou fait partie d'une expression, est en petites *capitales.* Le reste du texte encadre la lettrine. Il n'y a pas d'alinéa (retrait de paragraphe) à ce paragraphe.
Exemple

LOREM ipsum dolor sit amet, consectetuer adipiscing elit. Sed non risus. Suspendisse lectus tortor, dignissim sit amet, adipiscing nec, ultricies sed, dolor.

Lézarde

La lézarde est un défaut du *gris typographique* : c'est lorsque, sur plusieurs lignes successives, des espaces sont quasiment alignés verticalement, donnant une impression de zébrure, de fissure.

M

Majuscule

Les majuscules sont les initiales des phrases et des noms propres. Elles sont mises en grandes *capitales*, mais on peut avoir un texte en capitales sans que ce soient des majuscules, comme par exemple un titre en grandes capitales, ou un patronyme en petites capitales (Victor HUGO : seules le V et le H sont des majuscules).

Moins

Voir *tiret*.

P

Paragraphe

Un paragraphe est une portion de texte ayant une unité de sens (parlant d'un sujet donné par exemple, ou bien décrivant une scène, une situation). Il s'agit d'un découpage du texte permettant au lecteur de bien ordonner ses idées. Les autres découpages sont l'*alinéa* et la *phrase*. On a dans l'ordre hiérarchique : paragraphe > alinéa > phrase : un paragraphe comporte plusieurs alinéas et un alinéa comporte plusieurs phrases

On sépare deux paragraphes par une ligne vide, avec un retrait à la première ligne (retrait également appelé « alinéa »).

La plupart du temps, on confond les notions de paragraphe et d'alinéa ; en effet, si le découpage en chapitre, sections, sous-sections, … est assez « fin », il n'y a aucune nécessité de faire deux niveaux de distinction. Certains logiciels ont tendance à ne mettre que des paragraphes (par exemple Microsoft Word), d'autre à ne mettre que des alinéa (donc pas de saut de ligne au sein d'une section ou d'une sous-section) ; c'est la seconde solution qui est considérée comme correcte d'un point de vue typographique (la section ou la sous-section est composée d'un unique paragraphe qui contient plusieurs alinéas).

Parties d'un ouvrage

Un ouvrage comporte en début de volume, dans cet ordre :

- une *couverture* ;
- un feuillet blanc, ou s'il s'agit d'un ouvrage relié, deux feuillets blancs dont le premier sera collé à la couverture ;
- le *faux titre*, en *belle page* ;
- le *titre*, en belle page ;
- éventuellement la dédicace, en belle page ;
- éventuellement, un avis ou avertissement au verso *(fausse page)* d'un des feuillet précédent ;
- éventuellement, dans cet ordre et en belle page, l'*avant-propos*, l'*introduction* et la *préface* ;
- le texte proprement dit. La numérotation des pages débute à la première page de l'ouvrage, non à la première page du corps du texte.

Il comporte en fin de volume, dans cet ordre et en belle page :

- éventuellement la postface ;
- éventuellement les notes de fin d'ouvrage ;
- éventuellement la bibliographie ;
- éventuellement un ou des index, table des illustrations (avec le crédit photo) ;
- la table des matières ;
- le *colophon* ;
- l'achevé d'imprimé ;
- le dépôt légal et le nom de l'imprimeur ;
- un feuillet blanc, ou deux si l'ouvrage est relié.

On remarque qu'en typographie française, la table des matières se situe à la fin. Certains placent un sommaire (table des matières succincte) en début d'ouvrage, ou bien mettent la table des matières au début, à l'anglo-saxonne.

Les notes doivent figurer dans le même champ visuel que l'appel de note (soit dans la « marge » extérieure si on a la place, soit en pied de page).

Pica

Le pica est une longueur valant douze points anglo-saxons, soit 4,22 mm ou 0,17 pouces. Il y a donc environ 6 pica dans un pouce. Cette unité est appelée `pc` en CSS.

Point typographique

Le point est une unité de longueur utilisée en typographie. Il vaut à peu près 0,4 mm, il y a donc 2,5 points dans un millimètre. On distingue en fait :

- le point Didot de 0,375 9 mm (env. 3/8 mm, 1775) ; cette unité est appelée `dd` en LaTeX ;
- le point métrique de 0,4 mm (1790) ;
- le point de l'Imprimerie nationale, ou point IN, de 0,398 77 mm (en raison d'une erreur lors du passage au point métrique) ;
- le point anglo-saxon de 0,351 35 mm, soit 0,014 pouces ; il y a environ 72 points dans un pouce ; cette unité est appelée `pt` en CSS et dans LaTeX.

Voir aussi *cicéro* et *pica*.

Police

En typographie, la police est l'ensemble des caractères mobiles (pièces métalliques) suivant un dessin donné servant à réaliser l'impression. La police comprend les grandes lettres (grandes et petites *capitales*), les minuscules *(bas de casse)*, les lettres *ligaturées* et accentuées, et ce dans plusieurs *corps*, en *romain*, *italique* et en *gras*.

Exemple de polices : Arial, Bodoni, Chicago, `Courier`, Didot, Times, …

Dans le langage courant, la *police* est synonyme de fonte ; lorsque l'on veut faire la distinction, on parle de « famille de fonte » pour désigner la police.

T

Teletype

Teletype est un mot anglais signifiant « terminal ». Avant les années 1980, un terminal était un clavier et un écran reliés à un ordinateur distant, l'unité centrale ; plusieurs terminaux étaient donc reliés à cette unité centrale (celle-ci était très volumineuse). Du fait des faibles capacités d'affichage, l'écran pouvait contenir 25 lignes de 80 caractères, les lettre étaient toutes inscrites dans un rectangle de 8×8 pixels (les

pixels n'étaient pas carrés), on avait donc une police à chasse fixe. Le terme *teletype* est de fait souvent utilisé pour désigner les polices à chasse fixe.

Tiret

Les tirets sont des caractères qui sont formés d'un trait horizontal. Les typographes parlent de « moins » (bien que le signe mathématique moins soit différent). Les tirets sont différents du trait d'union ou de la *division*, ceux-ci étant plus courts. On distingue :

- le tiret *cadratin* ou tiret long « — » (caractère Unicode `U+2014`, HTML `&mdash;`, LaTeX `---`, clavier Mac Alt+-) ;
- le tiret *demi-cadratin* ou tiret moyen « – » (caractère Unicode `U+2013`, HTML `&ndash;`, LaTeX `--`, clavier Mac Alt+Maj+-).

En typographie française, le tiret demi-cadratin sert à introduire les éléments d'une liste non numérotée (en typographie anglaise, on utilise des *puces*), ou un dialogue. En typographie anglaise, il sert à indiquer séparer les deux nombres extrêmes d'un intervalle (par exemple 1938–2008) ; en typographie française, on utilise un trait d'union.

Le tiret cadratin ou demi-cadratin ou jouent un rôle similaire à une parenthèse (tiret d'incise, précédé et suivi d'une *espace* justifiante). Traditionnellement, on utilisait plutôt le tiret cadratin, mais celui-ci génère un blanc important ; certains préfèrent le tiret demi-cadratin pour mieux respecter le *gris typographique*.

Titre

Le titre figure sur le recto *(belle page)* du troisième feuillet suivant la couverture (dans le cas d'un ouvrage relié, le premier feuillet est collé à la couverture, le titre est donc sur le quatrième fuillet du cahier). Il comporte :

- le nom de l'auteur, ses titres et qualités ;
- le titre de l'ouvrage, et éventuellement le sous-titre ;
- éventuellement le nom du traducteur, du préfacier, de l'illustrateur, … ;
- le numéro de l'édition ;
- le nom de l'éditeur ou de l'imprimeur ;
- l'année de parution ;
- éventuellement, le nom de la collection.

Voir aussi *faux titre*, *parties d'un ouvrage*.

Trait d'union

Signe « - » séparant deux mots d'un mot composé. C'est le même caractère que la *division* qui marque la *césure*, mais le trait d'union est présent même en milieu de ligne. Il est plus court qu'un *tiret*.

Le caractère typographique a été conçu pour s'aligner au milieu de la lettre minuscule « x ». Avec les *capitales*, les typographes plaçaient le caractère la tête en bas pour qu'il apparaisse plus haut. En PAO ou avec un traitement de texte, il faut penser à décaler le caractère vers le haut pour qu'il soit centré sur la hauteur d'une capitale.

Il vaut mieux vérifier l'orthographe d'un nom composé dans le dictionnaire. Notons toutefois quelques règles générales :

- les nombres écrits en lettre inférieur à cent ont soit un trait d'union, soit un « et » : dix-sept, vingt et un, quatre-vingt-dix, mille neuf cent quatre-vingt-dix-sept ;
- les points cardinaux ont un trait d'union (nord-est) ;
- le trait d'union permet de distinguer un mot composé d'une expression de deux mots (par exemple « hors texte » et « un hors-texte ») ou d'un nom propre composé (Victor Hugo l'écrivain, mais le musée *Victor-Hugo*) ;
- un nom d'espèce individualisé est un nom propre composé d'un nom d'espèce indiquant le type d'objet qui est précisé par un autre nom ou un adjectif (par exemple dans « mont Blanc », « mont » est le nom d'espèce) ; lorsqu'un nom d'espèce individualisé sert lui-même à préciser un autre nom, on met un trait d'union (« le mont Blanc » mais « le massif du Mont-Blanc », « les îles du Cap-Vert ») ;
- les prénoms français composés prennent un trait d'union, y compris en abrégé (Pierre-Gilles de Gennes, P.-G. de Gennes) ; lorsque la personne utilise une partie comme prénom usuel, on la compose en italique (Louis-Charles-*Alfred* de Musset) ;
- les noms composés de rues et de villes françaises prennent un trait d'union, sauf pour l'article initial (avenue de la Porte-des-Lilas, rue Charles-de-Gaulle, La Rochelle, Saint-Étienne-du-Rouvray) ;
- lorsqu'une enseigne commerciale est reproduite partiellement, on met des traits d'union (par exemple « on va à la taverne *Au gosier en pente* » mais « le patron du Gosier-en-pente ») ;
- un mot composé avec « anti » ne prend pas de trait d'union sauf lorsque le second mot commence par « i » (anti-inflammatoire), que le second mot est lui-même composé (anti-sous-marin) ou lorsque le mot n'est pas courant mais formé pour la circonstance (anti-bruit) ;
- la plupart des mots composés avec « contre » prennent un trait d'union, mais il existe une cinquantaine d'exceptions (contrebande, contrebas, contrecarrer, contrecœur, contrefaçon, contrefort, contremaître, contrepartie, contrepétrie, contrepoids, contrepoint, contresens, contretemps, contrevenir, contrevérité, contrordre, …) ;
- les noms communs et les noms de ville composés avec « saint » prennent un trait d'union (un saint-honoré, saint Étienne désigne la personne ou le jour du calendrier, Saint-Étienne désigne la ville) ; certaines expressions prennent un trait d'union (Saint-Empire, Saint-Siège, Saint-Esprit, Sainte-Trinité, …) ;
- certains grades de l'armée comportent un trait d'union (ceux composés avec chef, amiral, colonel, major, lieutenant : sergent-chef, maréchal des logis-chef, sergent-major, …), les autres n'en ont pas (médecin général, maréchal des logis, premier maître, …).

En typographie française, le trait d'union sert aussi à séparer les extrêmes d'un intervalle (par exemple 1938-2008) ; les anglais utilisent le *tiret demi-cadratin* à la place.

Notes

1. P. Boman, C. Laucou, *La typographie cent règles*, éd. La Polygraphe, 2005, p. 75)

Voir aussi

- Orthotypographie (http://www.orthotypographie.fr/index.html) [archive], Jean-Pierre Lacroux

Index

Cette page est l'index des concepts. Pour l'index des caractères, commandes, environnements, classes de documents et extensions, voir *Commandes*.

Sommaire : Haut - A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A

B

C

D

E

Commandes

Cette page est l'index des caractères, commandes, environnements, classes de documents et extensions. Pour l'index des concepts, voir *Index*.

*****

- `{}` : voir *bloc*
- `\{` : 1
- `\}` : 1
- `\'` : 1
- ^ : voir *`\textasciicircum`*
- `\^` : 1
- \ : voir *`\textbackslash`*
- `\␣` (contre-oblique+espace) : voir *espace justifiante*
- `\#` : 1
- †, ‡ : voir *`\dag`*, *`\ddag`*
- `\$` : 1
- ␣ : voir *espace*, *`\textvisiblespace`*
- `\&` : 1
- `\`` : 1
- `` `! `` : 1
- `` `? `` : 1
- `\"` : 1
- £ : voir *`\pounds`*
- ø, Ø : voir *`\o`*, *`\O`*
- ¶ : voir *`\P`*
- § : voir *`\S`*
- · : voir *`\textperiodcenter`*
- … : voir *`\dots`*
- ¡ : voir *`` `! ``*, *`\textexclamdown`*
- ¿ : voir *`` `? ``*, *`\textquestiondown`*
- % : voir *commentaire*
- `\%` : 1
- ß : voir *`\ss`*
- `\_` : 1
- ~ : voir *espace insécable*, *`\textasciitilde`*
- `-` (-), `--` (–), `---` (—) : 1
- | : 1

  voir aussi *`\textbar`*

- `\,` : voir *espace, petite*
- `\/` : voir *espace fine*

A

- `abstract` (environnement) : 1, 2
- `\addcontentsline` : 1
- `\ae`, `\AE` : 1
- `aeguill` (extension) : 1
- `\appendix` : 1
- `apt-get` : 1
- `\arrayrulecolor` : 1
- `article` (classe) : 1
- `array` (extension) : 1, 2, 3, 4

B

- `babel` (extension) : 1, 2, 3, 4, 5, 6
- `\backmatter` : 1, 2, 3
- `\bigskip` : 1
- `breakbox` (environnement) : 1
- `\bsc` : 1, 2

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

X

Y

Liens externes

Voici quelques ressources en ligne disponibles:

En français

Communauté

- fr.comp.text.tex (news:fr.comp.text.tex) [archive] Groupe de discussion français à propos du (La)TeX et sur des sujets qui s'y rapportent (conseils d'utilisation (http://www.usenet-fr.net/fur/chartes/comp.text.tex.html) [archive], archives (http://groups.google.com/group/fr.comp.text.tex/) [archive])
- GUTenberg (http://www.gutenberg.eu.org/) [archive]
- « La francisation de LaTeX » (http://www.tug.org/tex-archive/language/french/french.html) (*Archive (https://web.archive.org/web/*/http://www.tug.org/tex-archive/language/french/french.html)* [archive] • *Wikiwix (https://archive.wikiwix.com/cache/?url=http://www.tug.org/tex-archive/language/french/french.html)* [archive] • *Que faire ?*)

Didacticiels et foires aux questions (FAQ)

- FAQ LaTeX de l'équipe Grappa (http://www.grappa.univ-lille3.fr/FAQ-LaTeX/) [archive] (Groupe de recherche en apprentissage automatique, Université de Lille).
- Le Framabook (livre libre) sur LaTeX *Tout ce que vous avez toujours voulu savoir sur LaTeX sans jamais oser le demander* (http://www.framabook.org/latex.html) [archive].
- Introduction au LaTeX (http://www.math-linux.com/spip.php?article52) [archive]
- « Une courte (?) introduction au LaTeX2e » (http://www.tug.org/tex-archive/info/lshort/french/flshort-3.20.pdf) (*Archive (https://web.archive.org/web/*/http://www.tug.org/tex-archive/info/lshort/french/flshort-3.20.pdf)* [archive] • *Wikiwix (https://archive.wikiwix.com/cache/?url=http://www.tug.org/tex-archive/info/lshort/french/flshort-3.20.pdf)* [archive] • *Que faire ?*)
- FRENCH-FAQ LATEX ON LINE (http://www.docsdunet.com/cours/faq-tex-french.html) [archive]
- TeX au collège (http://melusine.eu.org/syracuse/poulecl/) [archive]
- « Trucs et astuces sur LaTeX » (http://www.institut.math.jussieu.fr/~mpg/latex/tips) (*Archive (https://web.archive.org/web/*/http://www.institut.math.jussieu.fr/~mpg/latex/tips)* [archive] • *Wikiwix (https://archive.wikiwix.com/cache/?url=http://www.institut.math.jussieu.fr/~mpg/latex/tips)* [archive] • *Que faire ?*)
- Diapositives des conférences de formation à LaTeX (http://gte.univ-littoral.fr/members/dbitouze/pub/latex) [archive], D. Bitouzé

En anglais

Communauté

- The TeX Users Group (http://www.tug.org/) [archive] inclut des liens sur des versions libres de (La)TeX pour de nombreux types d'ordinateur.
- comp.text.tex (news:comp.text.tex) [archive] Groupe de discussion à propos du (La)TeX et sur des sujets qui s'y rapportent
- CTAN (http://www.ctan.org/) [archive] Des centaines de paquets et de programmes complémentaires

Didacticiels/FAQ (foire aux questions)

- Detexify (http://detexify.kirelabs.org/classify.html) [archive] un programme extraordinaire permettant de trouver des symboles LaTeX en les dessinant à la souris !!!
- Introduction au LaTeX (http://www.math-linux.com/spip.php?article52) [archive]
- Une courte (?) introduction au LaTeX2e (http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf) [archive], ou LaTeX2e en 131 minutes (fichier pdf de 2,16 MB)
- The UK TeX FAQ (http://www.tex.ac.uk/cgi-bin/texfaq2html?introduction=yes) [archive] Liste de questions et réponses fréquemment envoyées à comp.text.tex
- LaTeX Primer (http://www.maths.tcd.ie/~dwilkins/LaTeXPrimer/) [archive] Un guide vraiment utile
- The AMS Short Math Guide for LaTeX (http://www.ams.org/tex/short-math-guide.html) [archive], un sommaire concis des possibilités de composition de formules mathématiques
- « TeX on Mac OS X » (http://www.rna.nl/tex.html) (*Archive (https://web.archive.org/web/*/http://www.rna.nl/tex.html)* [archive] • *Wikiwix (https://archive.wikiwix.com/cache/?url=http://www.rna.nl/tex.html)* [archive] • *Que faire ?*) Guide d'utilisation de TeX et LaTeX sur un Mac
- Text Processing using LaTeX (http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/) [archive]
- The (La)TeX encyclopaedia (http://tex.loria.fr/index.html) [archive]
- Hypertext Help with LaTeX (http://www.giss.nasa.gov/latex/) [archive]
- Commands defined with * options (http://www.tex.ac.uk/cgi-bin/texfaq2html?label=cmdstar) [archive]

Référence

- Projet de site LaTeX (http://www.latex-project.org/) [archive]
- The Comprehensive TeX Archive Network (http://www.ctan.org) [archive] Le dernier logiciel de (La)TeX et ses paquets
- TeX Directory Structure (http://www.tug.org/tds/) [archive], utilisé par de nombreuses distributions (La)TeX
- Natural Math (http://www.math.missouri.edu/~stephen/naturalmath/) [archive] convertit le langage naturel en formules mathématiques

représentées en LaTeX
- Paquets et commandes obsolètes (http://www.ctan.org/tex-archive/info/l2tabu/english/l2tabuen.pdf) [archive]
- Version française du précédent (http://www.ctan.org/tex-archive/info/l2tabu/french/l2tabufr-light.pdf) [archive]
- Lamport's book *LaTeX: A Document Preparation System*
- LaTeX par la pratique (http://www.oreilly.fr/catalogue/2841770737.html) [archive] par Christian Rolland, chez O'Reilly France : pour bien débuter en français
- LaTeX (http://latex-pearson.org) [archive] par Denis Bitouzé & Jean-Côme Charpentier chez Pearson Education France (en français) : pour être guidé pas à pas depuis l'installation du système jusqu'à l'édition d'un document complet

Bibliographie

Typographie

- (français) *Lexique des règles typographiques en usage à l'Imprimerie nationale*, éd. Imprimerie nationale (2002), ISBN 2-7433-0482-0
- (français) P. Boman, C. laucou, *La typographie cent règles*, éd. La Polygraphe (2005), ISBN 2-909051-29-3
- (anglais) *The Chicago Manual of Style, 15th ed.*, éd. University of Chicago Press (2003), ISBN 0-226-10403-6

TeX

- (anglais) D. E. Knuth, *The TeXbook*, éd. Addison-Wesley (1984), ISBN 0-201-13447-0
- (français) T. Lachand-Robert, *La maîtrise de TeX et LaTeX*, éd. Masson (1995), ISBN 2-225-84832-7
- (français) P. W. Abrahams, K. A. Hargreaves, K. Berry, trad. M. Chaudemanche, *TeX pour l'impatient*, http://ctan.tug.org/tex-archive/info/impatient/fr/

LaTeX

- M. Goossens, F. Mittelbach, A. Samarin, *The LaTeX Companion*
  - (anglais) éd. Addison-Wesley (1993), ISBN 0-201-54199-8
  - (français) éd. CampusPress (2000), ISBN 2-7440-0897-4
- (anglais) F. Mittelbach, M. Goossens, J. Braams, D. Carlisle, C. Rowley, *The LaTeX Companion, 2nd edition*, éd. Addison-Wesley (2004), ISBN 0-201-36299-6
- (français) C. Rolland, *LaTeX par la pratique*, éd. O'Reilley (1999), ISBN 2-84177-073-7
- (français) B. Desgraupes, *LaTeX, apprentissage, guide et référence*, éd. Vuibert (2000), ISBN 2-7117-8658-7
- (français) D. Bitouzé & J.-C. Charpentier, *LaTeX, l'essentiel* (http://latex-pearson.org) [archive], éd. Pearson Education France, octobre 2010, 384 p. (ISBN 978-2-7440-7451-6).
- (français) D. Bitouzé & J.-C. Charpentier, *LaTeX : Synthèse de cours & Exercices corrigés* (http://latex-pearson.org) [archive], éd. Pearson Education France, 2006, 304 p. (ISBN 978-2-7440-7187-4) (édition revue et corrigée en mai 2008).
- (français) Collectif, *LaTeX pour l'impatient*, éd. H&K (2007), ISBN 2-35141-016-5