Fichier 4 : Tuto sur les injections Latex

Initiation à LATEX

guide-latex-fr

Pour débutants ou

jeunes utilisateurs

Par Adrien Bouzigues

Indignation 13 ‡215

13 juillet 2016

Version 3.7 à jour au

3 octobre 2021

Word

LATEX

Impossible à faire

Complexité et taille du rapport

Effort et temps requis

13

1

2

3

4

5

7 6

8

9

10

11

12

Cette œuvre, création, site ou texte est

sous licence Creative Commons Attribution - Pas d’Utilisation Commerciale -

Pas de Modification 4.0 International.

Pour accéder à une copie de cette licence, merci de vous rendre à l’adresse

suivante http://creativecommons.org/

licenses/by-nc-nd/4.0/ ou d’envoyer

un courrier à :

Creative Commons

444 Castro Street, Suite 900

Mountain View, California, 94041

USA

Toutes les versions de ce guide sont soumises

à cette licence Creative Commons, y compris

les plus anciennes qui peuvent circuler et qui

n’y font pas explicitement mention.

2

LATEX

Texte

(39 & 135)

Mise en

forme

Organiser

Page de

garde

Sommaire

En-têtes

et pieds

Tableaux

(82 & 173)

Tableaux

Boîtes

tcolorbox

Dessiner

(199 &

215)

PSTricks

TikZ

Maths

(66 & 170)

Formules

Alignement

Délimiteurs

Matrices

Images

(94 & 192)

Formats

Insertion

Montage

3

Sommaire

Préambule 8

I LATEX : histoire & premier contact 11

1 Pourquoi (utiliser) LATEX ? 12

1.1 Historique : de TEX à LATEX . . . . . . . . . . . . . . . . . . . 12

1.2 Pourquoi utiliser LATEX ? . . . . . . . . . . . . . . . . . . . . . 14

1.3 Autres arguments . . . . . . . . . . . . . . . . . . . . . . . . . 16

2 Installation de LATEX 18

2.1 Installation de MiKTeX . . . . . . . . . . . . . . . . . . . . . . 18

2.2 Installation de Texmaker . . . . . . . . . . . . . . . . . . . . . 20

2.3 Vérification finale . . . . . . . . . . . . . . . . . . . . . . . . . 20

3 Compiler avec LATEX 22

3.1 Principe de la compilation . . . . . . . . . . . . . . . . . . . . 22

3.2 Démarrer avec Texmaker . . . . . . . . . . . . . . . . . . . . . 23

3.3 Compiler avec Texmaker . . . . . . . . . . . . . . . . . . . . . 25

II Débuter avec LATEX 30

4 Les règles de base 31

4.1 Les règles pour faire du LATEX . . . . . . . . . . . . . . . . . . 31

4.2 Les 3 règles d’or en LATEX . . . . . . . . . . . . . . . . . . . . 32

4.3 La base d’un document LATEX . . . . . . . . . . . . . . . . . . 32

4.4 Les packages . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35

5 Gestion du texte et mise en forme 39

5.1 Notre premier texte . . . . . . . . . . . . . . . . . . . . . . . . 40

5.2 Un peu de mise en forme . . . . . . . . . . . . . . . . . . . . . 43

5.3 Organiser son document . . . . . . . . . . . . . . . . . . . . . 47

4

5.4 Gestion du sommaire . . . . . . . . . . . . . . . . . . . . . . . 49

5.5 La page de garde . . . . . . . . . . . . . . . . . . . . . . . . . 53

5.6 Création de commandes . . . . . . . . . . . . . . . . . . . . . 58

5.7 Les listes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60

5.8 Une petite touche de couleur ? . . . . . . . . . . . . . . . . . . 63

6 Les mathématiques sous LATEX 66

6.1 Le mode mathématiques . . . . . . . . . . . . . . . . . . . . . 67

6.2 Vers les espaces insécables . . . . . . . . . . . . . . . . . . . . 68

6.3 Des exemples de formules . . . . . . . . . . . . . . . . . . . . 70

6.4 L’affichage et les délimiteurs . . . . . . . . . . . . . . . . . . . 73

6.5 Les matrices . . . . . . . . . . . . . . . . . . . . . . . . . . . . 76

6.6 Aligner des équations . . . . . . . . . . . . . . . . . . . . . . . 78

7 Les tableaux et boîtes sous LATEX 82

7.1 Conventions . . . . . . . . . . . . . . . . . . . . . . . . . . . . 83

7.2 Création de tableaux . . . . . . . . . . . . . . . . . . . . . . . 84

7.3 Insérer une légende . . . . . . . . . . . . . . . . . . . . . . . . 86

7.4 Les boîtes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 87

7.5 Le Saint-Graal des boîtes . . . . . . . . . . . . . . . . . . . . . 92

8 Insérer des images 94

8.1 Les formats d’images . . . . . . . . . . . . . . . . . . . . . . . 95

8.2 Les longueurs . . . . . . . . . . . . . . . . . . . . . . . . . . . 96

8.3 Insérer une image . . . . . . . . . . . . . . . . . . . . . . . . . 98

8.4 Les références . . . . . . . . . . . . . . . . . . . . . . . . . . . 105

8.5 Un peu de montage . . . . . . . . . . . . . . . . . . . . . . . . 107

9 Traitement des erreurs 112

III Aller plus loin avec LATEX 118

Préambule – Le retour 119

10 Les moteurs de compilation sous LATEX 121

10.1 Présentation des différents moteurs . . . . . . . . . . . . . . . 121

10.2 Utilisation des différents moteurs de compilation . . . . . . . . 123

10.3 Bilan . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 125

5

11 Structurer ses documents 129

11.1 Un peu de rangement . . . . . . . . . . . . . . . . . . . . . . . 129

11.2 Commandes disponibles . . . . . . . . . . . . . . . . . . . . . 130

11.3 La pratique . . . . . . . . . . . . . . . . . . . . . . . . . . . . 131

11.4 D’autres solutions . . . . . . . . . . . . . . . . . . . . . . . . . 133

12 Améliorer son texte et sa mise en forme 135

12.1 Changer la police d’écriture . . . . . . . . . . . . . . . . . . . 135

12.2 Changer la taille de police . . . . . . . . . . . . . . . . . . . . 142

12.3 Inclure des fichiers PDF . . . . . . . . . . . . . . . . . . . . . 142

12.4 En-têtes et pieds de page . . . . . . . . . . . . . . . . . . . . . 145

12.5 Centrer verticalement du texte . . . . . . . . . . . . . . . . . . 150

12.6 Générer une bibliographie . . . . . . . . . . . . . . . . . . . . 152

12.7 Générer un index . . . . . . . . . . . . . . . . . . . . . . . . . 160

13 Mathématiques : remarques & astuces 170

13.1 Remarques générales . . . . . . . . . . . . . . . . . . . . . . . 170

13.2 Limites et indiçage . . . . . . . . . . . . . . . . . . . . . . . . 172

14 Tableaux & boîtes 173

14.1 Autres formats de cellules . . . . . . . . . . . . . . . . . . . . 173

14.2 Cellules centrées verticalement et horizontalement . . . . . . . 175

14.3 Fusion et coloriage de cellules . . . . . . . . . . . . . . . . . . 177

14.4 Créer sa propre boîte . . . . . . . . . . . . . . . . . . . . . . . 183

14.5 Afficher du code LATEX . . . . . . . . . . . . . . . . . . . . . . 186

15 Images : de nouvelles subtilités 192

15.1 Une référence toute prête . . . . . . . . . . . . . . . . . . . . . 192

15.2 Insérer des légendes intermédiaires . . . . . . . . . . . . . . . 193

15.3 Insérer un grand nombre de fichiers . . . . . . . . . . . . . . . 194

15.4 Insérer un fichier .svg . . . . . . . . . . . . . . . . . . . . . . 197

16 Dessiner avec PSTricks 199

16.1 Fonctionnement général . . . . . . . . . . . . . . . . . . . . . 199

16.2 Dessiner des circuits électriques . . . . . . . . . . . . . . . . . 200

16.3 Dessiner tout court . . . . . . . . . . . . . . . . . . . . . . . . 203

16.4 Utiliser des coordonnées . . . . . . . . . . . . . . . . . . . . . 206

16.5 Des boîtes pour le texte . . . . . . . . . . . . . . . . . . . . . 210

16.6 Réaliser des intersections . . . . . . . . . . . . . . . . . . . . . 211

16.7 Extraction du contour d’une image . . . . . . . . . . . . . . . 212

6

17 Dessiner avec TikZ 215

17.1 Démarrer sous TikZ . . . . . . . . . . . . . . . . . . . . . . . . 215

17.2 Un polygone régulier . . . . . . . . . . . . . . . . . . . . . . . 222

17.3 Automatiser les dessins . . . . . . . . . . . . . . . . . . . . . . 225

17.4 Dessiner des figures mathématiques . . . . . . . . . . . . . . . 231

17.5 Gestion des styles . . . . . . . . . . . . . . . . . . . . . . . . . 233

17.6 Insérer du texte . . . . . . . . . . . . . . . . . . . . . . . . . . 236

17.7 Création de graphes et de diagrammes . . . . . . . . . . . . . 242

17.8 Le mot de la fin . . . . . . . . . . . . . . . . . . . . . . . . . . 256

18 Faire des présentations avec Beamer 259

Annexes 260

7

Préambule

C

e guide a tout d’abord été construit pour mon usage personnel afin de

regrouper toutes mes connaissances en LATEX. Il sert aussi à mes camarades de promotion qui désirent se mettre à LATEX.

Accessoirement, dans l’éventualité où un parfait inconnu viendrait à lire

ce guide, j’espère qu’il pourra l’aider à son tour dans son initiation à LATEX.

D’autre part, mes connaissances en LATEX restent limitées. Je n’ai pas la

science infuse et ce guide est loin d’être exhaustif. Je propose juste des solutions qui fonctionnent. N’hésite donc pas à aller te documenter ailleurs

si un point ne te semble pas clair ou si tu cherches d’autres informations.

S’ils ne sont pas légion, il existe d’autres guides en français pour apprendre

le LATEX. Pour ma part, je recommande l’excellent LATEX. . . pour le prof de

maths ! d’Arnaud Gazagnes 1

, très complet et bien expliqué.

Je suis aussi tombé plus récemment sur Rédaction avec LATEX de Vincent

Goulet 2

, très agréable à lire et bien détaillé lui aussi.

Sur ce, bonne lecture !

Adrien Bouzigues

I13 Cl215

1. Disponible sur : http://math.univ-lyon1.fr/irem/spip.php?article340.

2. Disponible sur : https://ctan.org/pkg/formation-latex-ul.

8

Lien de mon site LATEX qui héberge le guide (sous la forme d’un

code QR généré par LATEX !)

TikZ Seal Of Quality

9

Ô Capitaine ! Mon Capitaine !

Pendant chaque traversée,

Tu restes à mes côtés

Et soutiens mon avancée.

10

Première partie

LATEX : histoire & premier

contact

11

Chapitre 1

Pourquoi (utiliser) LATEX ?

C

omme toute chose, LATEX possède une histoire qui lui est propre, des

avantages mais aussi des inconvénients – rien n’est parfait en ce monde.

Toutefois, LATEX est aussi un langage qui continue d’exister à l’heure actuelle

et qui reste une référence dans le milieu scientifique.

C’est pourquoi je te propose un petit interlude culturel avant d’entrer

dans le vif du sujet. . . et peut-être aussi pour finir de te convaincre de son

utilité !

1.1 Historique : de TEX à LATEX

La (petite) histoire

Donald Knuth est un mathématicien et informaticien américain, professeur émérite à l’université de Stanford. Il est l’auteur d’une bible de la

programmation intitulée The Art of Computer Programming (TAOCP). 1

Le premier volume paru en 1965, a été publié à l’ancienne avec des caractères en plomb. Quand en 1976 Donald Knuth décide de publier la seconde

édition du volume 2 de TAOCP, les caractères en plomb ont été abandonné

au profit de la photocomposition.

Donald Knuth trouve alors la qualité d’impression de ces machines, médiocre (notamment pour l’écriture des formules mathématiques) et décide de

créer deux logiciels pour pouvoir produire ses publications avec une qualité

typographique professionnelle.

1. Cet historique est extrait des « Fiches à Bébert », dont le texte complet est disponible

sur : http://lesfichesabebert.fr/divers/tex.html.

12

Chapitre 1. Pourquoi (utiliser) LATEX ?

;A<

Le premier, TEX, sert à la composition de documents ; le second, METAFONT, à produire des polices vectorielles. Donald Knuth va mettre plusieurs

années avant de sortir en 1983 la version définitive de TEX qui utilise la police

Computer Modern qu’il a créé à l’aide de METAFONT.

En effet, Donald Knuth s’était fixé comme but d’arriver à un produit

qui devrait être parfait et qui devrait le rester au cours du temps. C’est cette

version qui est toujours utilisée et qui fonctionne depuis 30 ans.

Donald Knuth est quand même intervenu sur TEX à plusieurs reprise,

notamment en 1989 pour l’adapter aux caractères nécessaires pour la composition de texte avec d’autres langues que l’anglais (version 2.991). La version

actuel de TEX est la 3.14159265 (janvier 2014).

L’autre trait de génie de Donald Knuth est de confier TEX à l’American

Mathematical Society et d’en faire un logiciel libre.

À partir de là, d’autres informaticiens vont s’emparer de TEXpour l’adapter (sortie de document au format PDF, utilisation de format d’image inconnue en 1983, adaptation à d’autres langues que l’anglais. . .) et l’enrichir

(module permettant la création de formule chimique, de partition musicale,

de diagrammes électrique ou physique. . .).

En 1982, Leslie Lamport, un chercheur en informatique américain, écrit

LATEX (Lamport TEX) un nouveau jeu de macros beaucoup plus simple à

utiliser que TEX.

C’est un succès et pratiquement plus personne n’utilise TEX. L’apparition

des packages, qui permettent facilement d’augmenter les fonctionnalités, ont

rendu LATEX incontournable (édition d’ouvrages scientifiques ou article de

recherches, notamment).

La version actuelle de LATEX est LATEX2ε, qui date de 1994. Elle est maintenue par le LATEX3 Project team qui nous prépare la version 3 de LATEX

depuis 20 ans !

À la fin des années 90, Hàn Thê Thành crée le moteur pdfTEX qui permet

de sortir les documents au format PDF, plus convivial que le format d’origine

de TEX le DVI.

La dernière version la 1.40.11 date de 2011. pdfTEX n’est plus développé,

seules des corrections de bug y sont apportées.

C’est ce moteur que nous allons utiliser par la suite, qui permet

de passer directement du fichier LATEX au fichier PDF final désiré.

13

Chapitre 1. Pourquoi (utiliser) LATEX ?

;A<

Étymologie et prononciation

Si je remercie encore une fois Bébert pour ce magnifique historique, je me

dois désormais d’intervenir sur un point qu’il ne traite pas sur cette page :

l’étymologie et la prononciation de LATEX.

C’est un point extrêmement crucial qui peut te permettre de briller lors

de soirées mondaines et d’éviter de passer pour un blaireau lors de conversations avec d’autres utilisateurs de LATEX.

De ce que j’ai lu un jour quelque part sur Internet, Donald Knuth a

nommé son logiciel TEX comme pour « technologie ».

Mais, il s’avère qu’il est aussi féru de grec. Et « technologie », en grec,

s’écrit « τεχνoλoγια », le χ correspondant au « chi » mais que l’on prononce

« khi ».

Et c’est donc pourquoi TEX se prononce « tech » mais s’écrit avec un “X”.

Quant à LATEX, il s’agit juste d’ajouter les premières lettres du nom de

son créateur, Leslie Lamport. TEX est donc devenu LATEX. . . et se prononce

a priori de la même façon.

Toutefois, Leslie Lamport indique explicitement dans son livre LaTeX :

A Document Preparation System qu’il n’encourage aucune prononciation particulière pour LATEX. . . mais là encore, si tu ne veux pas passer pour un blaireau, je t’encourage vivement à t’en tenir à la prononciation usuelle, soit

« latech » !

Bien, maintenant que ce point a été abordé, venons-en aux avantages à

utiliser LATEX avec, pour commencer, des témoignages !

1.2 Pourquoi utiliser LATEX ?

Durant l’été 2017, j’ai posé la question suivante sur le groupe « TeX /

LaTeX User Group » de LinkedIn :

La question posée

LaTeX professional experience

Hello everybody,

14

Chapitre 1. Pourquoi (utiliser) LATEX ?

;A<

I’m actually an engineering student and one of my main hobbies

is writing stuffs in LaTeX (scientifical reports, lessons’ synthesis,

letter. . ., even a LaTeX manual user for beginners (in French) !). I was

wondering if LaTeX is really helpful, in daily life, at work.

So, if anyone would like to share his opinion/experience, about how

he uses LaTeX at work (or not), feel free to answer my message.

Thanks a lot and have a good summer,

J’espère pour toi que l’anglais n’est pas une contrainte car c’est loin d’être

fini. Si toutes les réponses sont intéressantes, je trouve mon guide un peu terni

par 6 pages de commentaires. . . Je vais donc faire un petit résumé :

→ certains pensent qu’utiliser LATEX est pertinent uniquement dans un

milieu académique ou scientifique (recherche, surtout pour les mathématiques) ;

→ beaucoup travaillent avec des gens qui fonctionnent exclusivement sous

Word. Toutefois, pour la diffusion de notes internes, l’utilisation de

LATEX est appréciée (clarté du message, mise en page propre, simplicité. . .) ;

→ beaucoup reconnaissent que LATEX possède une forte courbe d’apprentissage, surtout au début 2

. Toutefois, ils utilisent aussi LATEX dans leur

quotidien (lettres, CV, rendus. . .) car ils préfèrent sa facilité d’utilisation par rapport à Word une fois l’apprentissage bien avancé ;

→ quasiment tous considèrent qu’apprendre à utiliser LATEX n’est pas une

perte de temps et peut se révéler utile.

Si tu n’es pas convaincu ou si tu crains que j’ai truqué les réponses, laissemoi au moins en partager deux, que tu puisses te faire une idée :

Les 2 réponses les plus pertinentes à mon sens

❖ Ed Blackburne : I use LaTeX everyday at work. My responsibilities include the production of Model Validation on reports per

2. Mais je te rassure, ce guide est justement conçu pour t’aider à passer ce cap difficile

15

Chapitre 1. Pourquoi (utiliser) LATEX ?

;A<

SR11-7. These are (generally) very technical and must be compliant

with our Enterprise standards as well as regulatory guidance. Although many of my colleagues use MS Word, my team enjoys increased productivity from LaTeX.

Additionally, for the econometric models, my team utilizes R/knitR/LaTeX to create dynamic reports (using methods borrowed

from reproducible research techniques).

I have created company-specific memo templates that I use on a

daily basis, as well.

If you write technical documents and/or need references (that work)

I highly encourage investing the minimal effort to become a competent LaTeX user.

❖ Brian Dunn : While LaTeX has a learning curve to use it well, so

does MS Word or LibreOffice Writer, many people never use a word

processor’s formatting “styles”, for example, and instead manually

format everything.

In talking with people at industrial trade shows, I occasionally

come across a company which uses LaTeX for their documentation.

Usually they are small engineering operations, and often European.

Most places use poorly-formatted MS-Word generated documentation, or else InDesign when they want a professional image. I

also found that companies which are suffering are not interested in

improving their documentation, sales literature, or websites, even

though their competitors which are doing well have very nice publicfacing literature.

Toujours pas convaincu ? Voici alors une ribambelle d’arguments qui devraient, j’espère, finir de te convaincre d’utiliser LATEX.

1.3 Autres arguments

Utiliser LATEX au lieu d’un autre logiciel de traitement de texte plus. . .

conventionnel présente un certain nombre d’avantages, dont voici la liste (non

exhaustive) :

→ LATEX est entièrement gratuit et utilisable sur n’importe quel système

d’exploitation ;

16

Chapitre 1. Pourquoi (utiliser) LATEX ?

;A<

→ un fichier LATEX est utilisable par n’importe qui (à condition d’avoir les

logiciels adaptés à LATEX) et sous n’importe quelle version de LATEX ;

→ LATEX génère un fichier PDF prêt à l’impression et lisible par n’importe

qui ;

→ LATEX propose une mise en page professionnelle et déjà paramétrée. La

gestion de la numérotation des pages, des en-têtes et des pieds de page

est relativement simple ;

→ écrire des formules mathématiques devient assez facile (avec un peu de

pratique) ;

→ LATEX gère intégralement les notes de bas de pages, les renvois, le sommaire, les images, les tableaux, les légendes et la numérotation, les

références bibliographiques ou la mise en place d’un index ;

→ LATEX réalise aussi les césures les plus appropriées et prend en compte

les ligatures.

Convaincu cette fois ? Pas vraiment ? Tu hésites encore ? Dans ce cas,

continuons sur notre lancée et installons LATEX sur notre ordinateur. Tu ne

peux pas savoir avant d’essayer, n’est-ce pas ?

17

Chapitre 2

Installation de LATEX

A

vant de commencer, je suppose que tu utilises un système d’exploitation

Windows. Dans le cas contraire, un utilisateur Linux devrait savoir se

débrouiller pour tout installer.

Si tu es un utilisateur d’Apple, je considère déjà ta cause perdue d’avance

et tu trouveras des équivalents grâce à Google. . . enfin, c’est ce que je disais

initialement. Désormais, les programmes que je présente par la suite te sont

aussi accessibles.

2.1 Installation de MiKTeX

MiKTeX est une distribution LATEX. Bon, je dois t’avouer que je ne sais pas

moi-même ce qu’est une distribution. . . Ce qui m’intéresse, c’est d’arriver à

faire fonctionner l’outil en question. Je vais donc sortir mon joker Wikipédia

pour cette fois :

Définition d’une distribution (informatique)

«

On parle souvent de distribution pour désigner un ensemble de

logiciels formant un tout cohérent et prêt à installer, incluant des

jeux de paquetages, le noyau du système d’exploitation, en particulier

le noyau Linux pour les distributions GNU/Linux (comme Debian,

Mandriva, Red Hat, Ubuntu, etc.), un système d’installation et des

utilitaires de configuration.

Cela désigne aussi un ensemble de paquets et d’outils utiles à la

création d’un document au format LaTeX et pour en faciliter l’utili18

Chapitre 2. Installation de LATEX

;A<

sation. Parmi les distributions LaTeX courantes, on trouve MiKTeX,

TeXLive, MacTeX2.

Par ailleurs, une base de données distribuées est répartie sur plusieurs nœuds, généralement sur différents serveurs.

»

Wikipédia – Disponible sur :

http://fr.wikipedia.org/wiki/Distribution#Informatique

Je ne sais pas si c’est plus clair ainsi. . . Ce qui est certain, c’est que le seul

élément intéressant à retenir est le suivant : MiKTeX est l’outil qui te permet

de transformer tes futures lignes de code LATEX en un PDF propre et lisible

par tous.

Pour installer MiKTeX, il faut procéder de la manière suivante 1

:

1) aller sur : http://miktex.org/download et télécharger l’exécutable ;

2) lancer l’exécutable et suivre les instructions d’installation ;

3) laisser les options par défaut DONT le “choix de poste” « Install

MiKTeX only for me ».

Pour débuter, elles conviennent parfaitement et le choix « only for

me » permet d’éviter tout problème par la suite.

!

Nota Bene

Je tiens à préciser que je n’ai aucun revenu financier grâce à

MiKTeX. Je conseille cette distribution car c’est celle que j’utilise et

qui fonctionne parfaitement pour ma part.

Elle a aussi l’avantage de proposer un gestionnaire de packages,

via MiKTeX Console ou l’interface de MiKTeX. Nous aurons l’occasion

d’y revenir plus tard dans ce guide, une fois que la notion de packages

aura été introduite.

1. Si besoin, un descriptif encore plus détaillé et imagé est disponible à l’adresse suivante : http://miktex.org/howto/install-miktex.

19

Chapitre 2. Installation de LATEX

;A<

!

En revanche, tu es libre de choisir la distribution de ton choix et

d’en prendre une autre. À toi d’en trouver une sur Internet : il y a

un peu de choix.

2.2 Installation de Texmaker

Techniquement, cette étape n’est pas nécessaire car tu pourrais écrire ton

fichier LATEX dans un fichier .txt (bloc-note) si le cœur t’en dit. Cependant,

le code sera plus compliqué à relire, il faut taper toutes les commandes à la

main et il faut indiquer à Windows – via des commandes dans le CMD – de

transformer ton code en PDF grâce à MiKTeX.

Avec Texmaker, tous ces tracas sont épargnés : tu as à disposition un

éditeur de fichiers LATEX performant, de la coloration syntaxique, un système

d’auto-complétion des formules fort pratique et agréable, et toutes les commandes pour utiliser MiKTeX sont intégrées et faciles à utiliser.

Pour cela, il faut aller sur le site de Texmaker : http://www.xm1math.

net/texmaker/download.html. Là encore, il suffit de télécharger l’exécutable, le lancer, suivre les instructions et laisser les options par défaut (comme

pour MiKTeX).

!

Nota Bene

Même remarque pour Texmaker que pour MiKTeX : tu peux choisir

un autre éditeur LATEX, même si celui-ci est vraiment très pratique

selon moi.

Il est aussi intégralement en français, avantage non négligeable

quand tu débutes.

2.3 Vérification finale

Si tu tiens à t’assurer que tout fonctionne, tu peux d’ores et déjà procéder

à une vérification finale comme décrit ci-après.

Si jamais tu rencontres le moindre problème, ne t’attarde pas

sur cette partie et poursuis au chapitre suivant, qui détaille l’utilisation des logiciels récemment installés.

1) Ouvrir Texmaker.

20

Chapitre 2. Installation de LATEX

;A<

2) En haut à gauche : Fichier puis Nouveau (ou Ctrl + N pour les

connaisseurs).

3) Recopier le code « Bonjour monde ! », fourni en-dessous, et sauvegarder

dans un dossier (le nom importe peu).

4) Appuyer sur F6 , attendre un peu, puis aller dans le dossier où tu

as sauvegardé le fichier : tu devrais y trouver un PDF avec la ligne

« Bonjour monde ! » écrite.

Bonjour monde !

\documentclass[]{report}

\begin{document}

Bonjour monde !

\end{document}

Tout fonctionne donc parfaitement ! Tu peux poursuivre sereinement la

suite du guide.

Dans le cas contraire, ne perds pas ton temps et passe directement à

la suite. Nous allons rapidement aborder le fonctionnement de Texmaker.

Si jamais des problèmes persistent par la suite, je ne peux que te conseiller

de tout désinstaller et de bien tout réinstaller comme indiqué précédemment.

21

Chapitre 3

Compiler avec LATEX

P

our faire du LATEX, il faut déjà connaître le point suivant : LATEX est un

langage et un système de composition de documents. Généralement, en

informatique, un langage requiert une étape obligatoire : la compilation. Et

LATEX n’échappe pas à cette règle.

3.1 Principe de la compilation

Quand tu vas rédiger un document sous LATEX, tu vas devoir procéder en

3 temps :

Code

PDF

Rédaction

(en LATEX)

Compilation

(moteur LATEX)

Document prêt

(fichier PDF)

Figure 3.1 – Les 3 étapes pour rédiger un document sous LATEX

Pour entrer un peu plus dans les détails, tu dois donc :

1) écrire ton document en LATEX (respect de ses conventions et utilisation

de commandes spécifiques) ;

2) demander à un moteur LATEX de transformer ton document et ses commandes en un fichier lisible et utilisable : c’est la compilation ;

22

Chapitre 3. Compiler avec LATEX

;A<

3) profiter du résultat fourni (format PDF) ou l’évaluer pour ensuite apporter des modifications au document, et ainsi de suite.

Quant au moteur LATEX utilisé, il en existe plusieurs. Pour débuter, je

recommande d’utiliser plutôt pdfTEX (intitulé apparemment PDFLATEX sous

Texmaker), qui permet de passer d’un coup du document LATEX au fichier

PDF final.

Quant aux autres moteurs, je les aborde bien plus loin dans ce guide,

en page 121. Je recommande plutôt de t’y rendre une fois que tu as un peu

d’expérience sous LATEX, pour ne pas perdre du temps et acquérir des bases

solides.

Nous savons désormais que nous devons compiler avec le moteur pdfTEX. . .

mais nous ne savons toujours pas comment faire ! Pas de panique : les logiciels

que je t’ai fait installer prennent tout en charge.

3.2 Démarrer avec Texmaker

Pour gérer et éditer ses fichiers LATEX, Texmaker est un excellent logiciel.

Et je sais de quoi je parle car, avant de m’y mettre, j’utilisais un autre logiciel,

tellement exécrable que j’ai fini par oublier son nom. Aujourd’hui, je ne fais

rien sans Texmaker. Voyons un aperçu de ce dernier :

Figure 3.2 – Aperçu de Texmaker

Revenons sur chaque point :

23

Chapitre 3. Compiler avec LATEX

;A<

→ barre de menus et d’outils : plein de commandes LATEX préremplies.

Personnellement, je l’utilise très rarement (y compris le bouton de sauvegarde). Je préfère utiliser les raccourcis clavier (en l’occurrence, Ctrl

+ S ) ;

→ structure du document : très pratique pour naviguer dans le code du

document ouvert ;

→ raccourcis de commandes : encore des commandes. Il peut être intéressant d’y jeter un coup d’œil une fois ce guide bien avancé. Il y a

principalement des commandes pour les formules mathématiques et

quelques unes pour la mise en forme du texte ;

→ code LATEX : c’est ici que tu tapes le texte de ton document et les

commandes LATEX nécessaires pour le mettre en forme ;

→ informations du compilateur : le résultat lors de la génération du PDF.

Très utile, s’il y a des erreurs, pour pouvoir se corriger ;

→ aperçu du PDF : une fenêtre avec l’aperçu du fichier PDF généré.

Si jamais cet aperçu n’est pas disponible (fenêtre d’affichage inexistante

comme sur mon image), il faut procéder de la manière suivante :

1) Aller dans Options puis dans Configurer Texmaker.

2) Dans l’onglet Afficheur Pdf, choisir les options Afficheur Pdf interne

et Intégré à la fenêtre. Valider.

3) Un bouton Pdf Viewer est alors disponible en bas à gauche et te permet d’activer ou non cette fenêtre d’aperçu.

Si jamais ce n’est pas clair, j’espère que cette capture d’écran permettra

de lever le moindre doute :

24

Chapitre 3. Compiler avec LATEX

;A<

Figure 3.3 – Procédure pour obtenir l’aperçu du PDF

L’aide en ligne

Si jamais tu as d’autres questions sur Texmaker, son site officiel (http://www.xm1math.net/texmaker/index\_fr.html) est le

meilleur endroit pour avoir des réponses et fournit aussi un tutoriel

pour débuter avec LATEX.

Si certains points de ce guide te semblent obscures, tu peux donc

t’y rendre, ainsi que sur http://fr.wikibooks.org/wiki/LaTeX.

N’hésite pas quand tu débutes. Les deux sites sont en français et

répondent à beaucoup de questions assez facilement (i.e. avec un code

simple).

3.3 Compiler avec Texmaker

Maintenant que l’environnement propre à Texmaker a été présenté, voyons

un peu plus dans le détail un dernier point : la compilation. Pour commencer,

reprenons le code « Bonjour monde ! » utilisé en page 21 :

25

Chapitre 3. Compiler avec LATEX

;A<

Bonjour monde !

\documentclass[]{report}

\begin{document}

Bonjour monde !

\end{document}

Je suppose que tu as suivi les premières indications fournies, soit ouvrir

Texmaker, recopier le code « Bonjour monde ! » donné et enregistrer ton

document.

Si tu n’as pas précisé d’extension, tu remarqueras au passage que ton

fichier a été sauvegardé avec l’extension .tex, qui correspond à l’extension

pour des fichiers LATEX.

Il existe ensuite 3 façons de lancer la compilation de ton document LATEX :

→ via l’invite de commandes de ton système d’exploitation (le CMD pour

les utilisateurs de Windows). . . mais je n’en parlerai pas pendant ce

guide.

Sache cependant que c’est possible mais ne présente aucun intérêt

comme Texmaker propose des solutions plus pratiques ;

→ via Texmaker avec des clics souris ;

→ via Texmaker avec des raccourcis clavier (le plus rapide à mon sens).

Revenons sur les 2 derniers points plus dans le détail, pour que tu comprennes bien les actions à effectuer.

Pour une compilation via Texmaker avec des clics souris, il faut procéder

en 3 temps (cf. Figure 3.4 si besoin) :

1) Choisir le moteur de compilation, PDFLATEX dans notre cas, en haut

dans la barre d’outils.

2) Lancer la compilation en cliquant sur la flèche à gauche du choix du

moteur de compilation. Attendre que la compilation soit terminée.

26

Chapitre 3. Compiler avec LATEX

;A<

3) Juste à droite du choix du moteur de compilation, bien choisir l’option

Voir PDF et cliquer sur la flèche associée pour afficher le résultat.

Figure 3.4 – Lancer la compilation avec des clics souris (Texmaker)

Pour une compilation via Texmaker avec des raccourcis clavier, il faut

procéder en 2 temps (cf. Figure 3.5 si besoin) :

1) Lancer la compilation avec le moteur PDFLATEX avec la touche F6 .

2) Afficher le résultat avec la touche F7 .

Ces raccourcis sont personnalisables dans les options de Texmaker, comme

décrit ci-après (Figure 3.5).

Figure 3.5 – Configuration des raccourcis clavier pour lancer la compilation

(Texmaker)

27

Chapitre 3. Compiler avec LATEX

;A<

Mais il y a encore plus rapide : lancer la compilation ET avoir l’aperçu

du PDF actualisé en un seul raccourci clavier. C’est ce que Texmaker appelle

la « compilation rapide » 1

.

Tout d’abord, il faut s’assurer que la compilation rapide est bien programmée. Pour ce faire, il faut configurer Texmaker de la manière suivante :

Figure 3.6 – Configuration de la compilation rapide (Texmaker)

Ensuite, il suffit d’appuyer sur la touche F1 pour lancer la compilation rapide. Il s’agit de la touche par défaut, paramétrable dans les options

Texmaker comme indiqué en Figure 3.5.

Enfin, pour terminer les explications, il faut savoir que les compilations

réalisées sous Texmaker sont équivalentes à l’utilisation de l’invite de commandes. Grâce à Texmaker, cette utilisation est transparente et grandement

simplifiée. . . pour les non-initiés nous dirons !

!

La compilation : le conseil personnel

Peu importe le moteur utilisé pour la compilation, tu peux

remarquer qu’un fichier .tex entraîne toujours la génération d’autres

fichiers. C’est pourquoi je recommande toujours de travailler

avec le fichier .tex placé dans un dossier, pour éviter de

submerger tes autres dossiers et de te perdre parmi les fichiers.

1. Il s’agit du nom attribué par le concepteur de Texmaker. Le temps nécessaire pour

compiler le document n’est en rien diminué.

28

Chapitre 3. Compiler avec LATEX

;A<

!

Enfin, ces fichiers secondaires n’ont besoin d’être conservés que

le temps de travailler sur un document LATEX. Le seul fichier qui

compte est celui avec l’extension .tex. C’est lui qui contient

tout le code nécessaire à la compilation et à l’obtention du PDF final.

Attaquons désormais la raison première de ce guide : faire du LATEX.

29

Deuxième partie

Débuter avec LATEX

30

Chapitre 4

Les règles de base

4.1 Les règles pour faire du LATEX

Ces règles sont officieuses : je les ai élaborées à partir de mon expérience

personnelle avec LATEX. Elles restent donc pragmatiques et peuvent paraître

un peu farfelues mais sont importantes à mes yeux :

Les 5 règles pragmatiques

Règle no 1 : Tout est possible en LATEX

a

.

Règle no 2 : La règle no 1 est toujours vraie.

Règle no 3 : LATEX implique d’écrire des commandes soit des lignes

de code. Aérer et ordonner son code en facilite la relecture.

Règle no 4 : La voie de la perfection en LATEX passe par une recherche régulière sur Internet.

Règle no 5 : Si tu rencontres des difficultés, il ne faut pas hésiter à

demander des conseils.

a. Y compris écrire des partitions de musique : http://fr.wikibooks.org/

wiki/LaTeX/%C3%89crire\_de\_la\_musique

Passons maintenant sur des règles plus concrètes vis-à-vis de l’écriture

d’un code LATEX.

31

Chapitre 4. Les règles de base

;A<

4.2 Les 3 règles d’or en LATEX

Pour écrire du code LATEX, il existe 3 règles, suffisamment importantes à

mon sens pour être en or :

Les 3 règles d’or en LATEX

Règle d’or no 1 : Toute commande LATEX débute par un backslash

“\”.

Windows : Alt Gr + 8

Apple : Alt + + /

Règle d’or no 2 : Tout texte concerné par une commande LATEX est

délimité par des accolades “{” et “}”.

Windows : Alt Gr + 4 et Alt Gr + =

Apple : Alt + ( et Alt + )

Règle d’or no 3 : Toute commande LATEX qui comprend un begin

finit par un end.

Ce genre de structure s’appelle un environnement.

Il s’agit donc, selon moi, de la base pour écrire du code LATEX. Respecter

ces règles permet d’éviter un bon nombre d’erreurs, nombreuses quand tu

débutes.

Ces 3 règles prendront leur sens sous peu, quand nous allons mettre en

forme notre document et commencer à faire du LATEX (cf. 5.2 Un peu de mise

en forme, p. 43).

4.3 La base d’un document LATEX

Pour commencer, démarrons un fichier LATEX : ouvrons Texmaker, créons

un nouveau fichier et enregistrons-le au format .tex 1

.

1. Pour information/rappel, un fichier LATEX possède toujours l’extension .tex

32

Chapitre 4. Les règles de base

;A<

!

Codes LATEX fournis

Tout au long de ce guide, des exemples de code LATEX

sont fournis dans des encadrés verts clairs. Ils ont été testés

par mes soins avec le moteur PDFLATEX : tout devrait donc

fonctionner aussi de ton côté.

Toutefois, la copie du code depuis ce guide au format PDF semble

encore présenter quelques lacunes : saut de ligne lors d’une coupure

(ligne de code trop longue), apostrophe différente de celle présente

sous Texmaker. . .. Des erreurs lors de la génération du document

PDF peuvent alors survenir.

À toi de voir si tu préfères recopier chaque ligne de code – ce

qui facilite la mémorisation et l’apprentissage selon moi – ou si tu

préfères copier-coller et habilement utiliser la fonction « Remplacer »

de Texmaker.

La base d’un document LATEX est la suivante :

La base d’un document LATEX

\documentclass[options]{classe}

% Préambule

\begin{document}

% Ici s'écrit notre texte

% Notons que le symbole "%" permet de mettre un commentaire

\end{document}

Tout ce que j'écris après \end{document} n'a aucun intérêt et

ne sera pas interprété par LaTeX.

Plusieurs points importants sont à retenir :

→ \documentclass permet de définir le type de document (appelé « classe »

en LATEX) sur lequel tu vas travailler ;

33

Chapitre 4. Les règles de base

;A<

→ la zone entre \documentclass et \begin{document} s’appelle le préambule. Je décris cette partie en 4.4 Les packages, p. 35 ;

→ un premier exemple d’illustration de la règle d’or no 3 : un begin implique un end.

Seuls le texte et les commandes LATEX écrits entre \begin{document} et

\end{document}, hormis les commentaires, sont interprétés par LATEX

lors de sa création du fichier PDF final ;

→ tout ce qui peut être écrit après \end{document} n’est pas pris en

compte par LATEX.

Que mettre maintenant dans la ligne \documentclass ? Il s’agit ici de

définir le type de document à mettre en forme. En LATEX, le terme consacré

est « classe ». Définir la classe d’un document LATEX revient à utiliser un

gabarit spécifique pour le document, défini par défaut dans le code source de

LATEX, et entièrement personnalisable par la suite si besoin.

Il existe plusieurs classes, à renseigner à l’endroit où il y a écrit classe

dans mon exemple générique : report pour taper des rapports ; article

pour des articles scientifiques ; book pour des livres et letter, tu as compris

je pense, pour des lettres.

La partie options permet ensuite de renseigner toutes les options propres

à une classe. Les plus communs sont la taille du papier (A4 : a4paper, A5 :

a5paper) ainsi que la taille de police de base (10pt, 11pt ou 12pt). Mais il

en existe d’autres, en fonction des classes utilisées.

Personnellement, je recommande de commencer un nouveau document par \documentclass[a4paper, 12pt]{report}. Ce choix convient

pour 90 % des cas : ainsi, le risque de problème est moindre. Toutefois, si

tu as un petit rapport d’une vingtaine de pages à rédiger, la classe article

constitue aussi un choix judicieux.

Néanmoins, il faut bien garder à l’esprit que d’autres options de présentation de document existe (book, article ou letter). Ces derniers peuvent

toujours servir.

34

Chapitre 4. Les règles de base

;A<

Pour aller plus loin

Des explications plus précises et poussées (toutes les options de report, article, etc.) sont disponibles à http:

//fr.wikibooks.org/wiki/LaTeX/Les\_classes.

Ce qu’il faut bien comprendre et surtout retenir, c’est que la forme

finale de ton document est intrinsèquement liée à sa classe et

aux options choisies.

!

Une question ?

« Et si je veux changer la police en 14pt, comment faire ? »

Ah, je vois que le fond de la classe suit. J’aborde ce point en 5.2

Un peu de mise en forme, p. 43.

C’est bon ? Toujours là ? Tu verras, avec de la pratique, les bases vont rentrer. Plus qu’un dernier point un peu théorique à aborder et nous passerons

à la pratique. Promis !

4.4 Les packages

Si le lecteur curieux ne s’est pas encore empressé de faire des essais, je lui

recommande d’essayer le code suivant : 2

Un premier essai

\documentclass[a4paper, 12pt]{report}

\begin{document}

J'aime écrire en \LaTeX{} !

\end{document}

Si jamais tu ne sais pas quoi faire du code, je ne peux que t’inviter à te

2. Les sauts de ligne sont importants pour la lisibilité (règle pragmatique no 3).

35

Chapitre 4. Les règles de base

;A<

rendre en page 22. Tu y trouveras tout un chapitre consacré à la compilation

sous LATEX, soit l’étape pour transformer ton code en un PDF !

Normalement, suite à la compilation, tu as dû obtenir :

J’aime crire en LATEX !

Analysons le résultat. La règle d’or no 1 commence à prendre du sens :

une commande LATEX commence par un backslash “\” (ou contre-oblique

pour les puristes). Cette commande me permet d’écrire le mot « LaTeX »

d’une manière plus élégante.

Par contre, aucune trace du “é”. C’est bizarre : moi j’arrive à l’écrire sans

souci ! C’est parce que tu n’as pas dit à LATEX d’écrire en UTF-8 3

!

Pour ce faire, il faut dire à LATEX de charger des options supplémentaires.

Dans le jargon LATEX, ces options sont appelées des packages. Dans la littérature française, le terme de « paquetages » est parfois employé.

Les packages sont toujours renseignés dans le préambule, soit

entre les lignes \documentclass[options]{classe} et \begin{document}.

Pour charger un package, il faut utiliser la commande :

\usepackage[options\_du\_package]{nom\_du\_package}

Pour rédiger des documents en français, il est recommandé de remplir le

préambule de la manière suivante :

Un exemple qui fonctionne bien

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern} % Police standard sous LaTeX : Latin

Modern

% (alternative à la police d'origine développée par Donald

Knuth : Computer Modern)

\usepackage[french]{babel} % Pour la langue fran¸caise

\usepackage[utf8]{inputenc} % Pour l'UTF-8

\usepackage[T1]{fontenc} % Pour les césures des caractères

accentués

3. L’UTF-8 est un codage de caractères informatiques, qui tolère les accents : http:

//fr.wikipedia.org/wiki/UTF-8.

36

Chapitre 4. Les règles de base

;A<

\begin{document}

J'aime écrire en \LaTeX{} !

\end{document}

Je sens la curiosité briller dans ton regard donc je vais essayer de te donner

un peu plus de détails que les commentaires fournis 4

:

❖ le package inputenc, avec l’option utf8, permet de prendre en compte

l’utilisation de caractères accentués dans le fichier source (soit ton fichier .tex). Concrètement, inputenc se contente en fait de faire luimême la conversion entre les caractères accentués et les commandes

d’accentuation propres à LATEX ;

❖ si inputenc gère l’affichage des caractères accentués, la césure reste catastrophique ! Pour indiquer au compilateur les règles de césure pour les

mot accentués, il faut donc utiliser le package fontenc, avec l’encodage

T1 en option ;

❖ là encore, le résultat est loin d’être parfait. Le fait de charger fontenc

remplace les polices par défaut par des fontes de type 3, c’est-à-dire

non vectorielles. En clair, si tu zoomes sur un caractère accentué de

ton PDF, il sera pixélisé. D’où le chargement en amont de la police

Latin Modern, via le package lmodern ;

❖ enfin, pour s’adapter à la langue de Molière, le package babel avec

l’option french est indispensable !

!

Une question ?

« Pourquoi dire à LATEX d’aller chercher des options alors que

rien n’a été précisé pour la commande \LaTeX{}, par exemple ? »

Tout simplement parce qu’il s’agit d’une commande présente

4. Les explications qui suivent proviennent du site http://blog.dorian-depriester.

fr/latex/mais-a-quoi-bon-servent-les-packages-fontenc-et-inputenc

37

Chapitre 4. Les règles de base

;A<

!

de base dans le code source de LATEX. Il n’y a donc pas besoin de

charger quoi que ce soit au préalable.

Sache aussi que les packages sont construits par les utilisateurs

LATEX. C’est pourquoi tout est possible avec LATEX : tout est modifiable ou n’attend qu’à être créé.

C’est bon ? Toujours de la partie ? Dis-toi que, désormais, tu vas enfin

pouvoir écrire des paragraphes. Passons donc à la pratique !

38

Chapitre 5

Gestion du texte et mise en

forme

M

aintenant que nous connaissons les règles de base pour faire du LATEX

et que les packages ont été introduits, nous allons pouvoir commencer

à écrire du texte sous LATEX.

Par la suite, pour alléger les exemples, le préambule ne sera plus renseigné

dans les codes LATEX mis à disposition. Ces derniers seront basés sur l’architecture du code minimal fourni ci-après. L’ajout de nouveaux packages sera

signalé au début du code par un commentaire.

Le code minimal

\documentclass[a4paper, 12pt]{report}

% PDFLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\begin{document}

% Ecrire le code ici !

\end{document}

39

Chapitre 5. Gestion du texte et mise en forme

;A<

5.1 Notre premier texte

Prêt à enfin écrire un roman ? Bon, nous allons attaquer en douceur. Et,

petit à petit, tu auras suffisamment d’outils à ta disposition pour profiter des

fonctionnalités offertes par LATEX. Poursuivons avec le code suivant :

Un nouvel essai

J'aime écrire en \LaTeX{} ! Vraiment ! Surtout avec des

phrases longues qui prennent de la place. % Saut de ligne

Et toi ? Qu'en est-il ? \\ % Beaucoup d'espace et un

nouveau symbole (\\)

Pardon ? Tu débutes ? Tu vas voir, c'est facile. % Double

saut de ligne

Tu devrais normalement obtenir :

Le résultat

J’aime écrire en LATEX ! Vraiment ! Surtout avec des phrases longues

qui prennent de la place.

Et toi ? Qu’en est-il ?

Pardon ? Tu débutes ? Tu vas voir, c’est facile.

Nous pouvons relever plusieurs points :

→ un saut de ligne à l’écran est interprété comme un retour à la ligne ;

→ la commande \\ permet un véritable saut de ligne et donc de créer un

nouveau paragraphe ;

→ les espaces et saut de ligne intempestifs ne sont pas pris en compte ;

→ les alinéas sont automatiques (pas besoin de faire de tabulations).

Contrairement à Word, à première vue, LATEX possède une façon un peu

curieuse d’aller à la ligne ou de faire un saut de ligne (nouveau paragraphe).

40

Chapitre 5. Gestion du texte et mise en forme

;A<

Mais rappelons que la raison d’être de LATEX est de séparer le fond de la

forme : nous tapons le fond pour laisser LATEX le mettre en forme lors de la

compilation.

Mais l’utilisateur garde le contrôle et peut influer sur la forme grâce à

des commandes LATEX. C’est bien ce qui se passe ici. Si l’utilisateur veut un

retour à la ligne, il doit sauter une ligne dans son code. S’il veut sauter une

ligne, il doit utiliser la commande \\ et sauter une ligne dans son code.

Nous avons aussi pu remarquer que LATEX gère tous les problèmes liés

à l’espacement entre les mots. Il est donc inutile de faire un grand nombre

d’espaces ou de saut de ligne pour aérer son texte. Encore une fois, ce n’est

pas la politique de LATEX et il faut passer par des commandes si besoin.

Si le saut de ligne est disponible grâce à la commande \\ – qui, au passage, est cumulable –, l’utilisateur peut jouer sur l’espacement vertical grâce

à la commande \vspace{longueur}, avec v pour vertical et space pour espace. Il en va de même pour un espacement horizontal avec la commande

\hspace{longueur}.

La longueur est totalement libre, à condition de renseigner correctement

l’unité : 13mm ou 215pt, par exemple. Concrètement, nous pouvons procéder

de la manière suivante :

Gérer l’espacement

J'aime toujours écrire en \LaTeX

{}.

\vspace{1cm}

Surtout \hspace{8mm} quand je

laisse du blanc !

J’aime toujours écrire en

LATEX.

Surtout quand je

laisse du blanc !

Si j’ai rapidement annoncé qu’il était possible de cumuler la commande

\\ pour engendrer la création de plusieurs sauts de ligne, il existe aussi une

longueur 1

définie nativement sous LATEX et qui correspond à un saut de ligne.

Cette longueur est disponible grâce à la commande \baselineskip. Voyons

son utilisation sur un cas pratique :

1. Nous aurons l’occasion de revenir sur ce point en page 96.

41

Chapitre 5. Gestion du texte et mise en forme

;A<

Sauts de ligne et longueur baselineskip

Il est possible de sauter

plusieurs \\ \\ % Double saut

de ligne

lignes \\ \\ \\ % Triple saut de

ligne

ainsi.

\vspace{2\baselineskip} % Double

saut de ligne

Cette solution est aussi possible

, tout comme celle-ci ! \\[

\baselineskip] % Double saut

de ligne

Bref, beaucoup de fa¸cons de

sauter des lignes, de manière

plutôt concise.

Il est possible de sauter

plusieurs

lignes

ainsi.

Cette solution est aussi

possible, tout comme

celle-ci !

Bref, beaucoup de façons

de sauter des lignes, de

manière plutôt concise.

Enfin, sache qu’il est possible de rentrer des valeurs négatives, comme

-13mm ou -215pt. C’est surtout pratique pour remonter du texte lors de

montages, voire des images si besoin. Je recommande juste de limiter cette

pratique : tu risques de perdre beaucoup de temps à ajuster ton document.

!

Une question ?

« Que se passe-t-il si je vais juste à la ligne dans mon code LATEX ? »

Rien. Seul un saut de ligne à l’écran compte comme un retour à

la ligne.

Cependant, pour obtenir un retour à la ligne sur ton document, tu

peux aussi terminer ta phrase par la commande \\ et faire un simple

retour à la ligne dans ton code.

Bon, maintenant que nous avons toutes les cartes en main pour écrire des

paragraphes, passons à de la mise en forme.

42

Chapitre 5. Gestion du texte et mise en forme

;A<

5.2 Un peu de mise en forme

Comme je l’ai déjà annoncé, avec LATEX, tu rédiges le fond et lui laisses

le soin de s’occuper de la forme. Si nous en avons eu un premier aperçu, tu

vas pouvoir t’en rendre véritablement compte dès à présent.

Si tu veux mettre un texte en gras ou en italique, il faut donc l’indiquer

à LATEX par le biais de commandes bien spécifiques :

Gras & italique

\textbf{texte en gras} \\

\textit{texte en italique}

texte en gras

texte en italique

Comme tu peux le constater, tu écris la commande – qui débute par un

backslash – et tu encadres le texte concerné par des accolades. Fort heureusement, LATEX ne propose pas uniquement le gras et l’italique :

Table 5.1 – Les différentes possibilités de mise en forme du texte

Texte Rendu Environnement

\textbf{gras} gras bfseries

\textit{italique} italique itshape

\emph{emphase} emphase em

\textsl{penché} penché slshape

\textsc{Petites Capitales} Petites Capitales scshape

\textsf{sans empattement} sans empattement sffamily

\texttt{machine} machine (à écrire) ttfamily

Je vais revenir plus en détail sur l’emphase, avec la commande \emph{texte},

qui ne correspond pas à de l’italique.

En typographie, l’emphase permet d’accentuer un mot ou une phrase

grâce à un style ou une police différente de celle du reste du texte. Essayons

avec un exemple répandu de faux texte : le lorem ipsum 2

.

2. Pour plus de renseignements : http://fr.wikipedia.org/wiki/Faux-texte et

http://fr.lipsum.com/.

43

Chapitre 5. Gestion du texte et mise en forme

;A<

L’emphase ou emphasis

\textbf{Lorem ipsum dolor sit

amet, consectetur adipiscing

elit. \emph{Nunc est leo,

facilisis non nisi eget,}

auctor eleifend metus.} \\

\textit{Vestibulum porttitor,

ligula vitae suscipit

bibendum, \emph{lorem ligula

vestibulum ipsum,} sed

ultricies tellus dolor sit

amet odio.}

Lorem ipsum dolor

sit amet, consectetur

adipiscing elit. Nunc

est leo, facilisis non

nisi eget, auctor eleifend metus.

Vestibulum porttitor, ligula vitae suscipit bibendum, lorem ligula vestibulum ipsum, sed ultricies tellus dolor sit amet

odio.

Comme tu peux le constater, LATEX a adapté la mise en forme du texte

avec emphase en fonction de la mise en forme du paragraphe !

Voyons maintenant sur un exemple comment augmenter ou réduire la

taille de la police, ainsi que les notes de bas de page :

Taille de police et note de bas de page

% Différentes tailles de texte

{\tiny tiny} {\scriptsize

scriptsize} {\footnotesize

footnotesize} {\small small}

{\normalsize normalsize} {

\large large} {\Large Large}

{\LARGE LARGE} {\huge huge} {

\Huge Huge} \\

% Note

Note de bas de page\footnote{La

note en question.}.

tiny scriptsize footnotesize

small normalsize large

Large LARGE

huge Huge

Note de bas de page a

.

a. La note en question.

Comme tu peux le constater, LATEX gère lui-même la numérotation des

notes de bas de page, sans rien avoir besoin de lui indiquer. Pratique, n’est-ce

pas ?

44

Chapitre 5. Gestion du texte et mise en forme

;A<

Si jamais tu as besoin d’appliquer une mise en forme ou une taille de

police à plusieurs paragraphes, LATEX ne saura pas interpréter le changement

de paragraphe. Dans ce cas, il faut passer par un environnement 3

:

Mise en forme et taille – Environnement

\begin{Large}

Paragraphe 1.

Paragraphe 2. \\

\end{Large}

\begin{bfseries}

Paragraphe 3.

Paragraphe 4.

\end{bfseries}

Paragraphe 1.

Paragraphe 2.

Paragraphe 3.

Paragraphe 4.

LATEX propose aussi un moyen très simple pour accentuer les majuscules.

Il suffit d’utiliser un backslash, suivi de l’accent désiré. Puis, tu écris ton mot

normalement, avec une majuscule.

Accentuation

% Accent sur les majuscules

\'E, \`E, \^E et \c{C} \\

% "o pris dans e"

\OE{}il, c\oe{}ur

É, È, Ê et Ç

Œil, cœur

Il existe aussi des commandes spécifiques pour des symboles fréquemment

utilisés. Je pense notamment aux guillemets et aux points de suspension. La

preuve par l’exemple :

3. Pour la mise en forme du texte, utiliser les noms définis dans la colonne « Environnement » de la Table 5.1.

45

Chapitre 5. Gestion du texte et mise en forme

;A<

Autres symboles utiles

% Guillemets

\og guillemets fran¸cais \fg{} et

``guillemets anglais'' \\

% Points de suspension

Points de suspension\dots{} \\

% Tirets

Tiret court : - \\

Tiret moyen : -- \\

Tiret long : --- \\

% Esperluette - Pourcentage

\& et \textit{\&} ; \%

« guillemets français » et

“guillemets anglais”

Points de suspension. . .

Tiret court : -

Tiret moyen : –

Tiret long : —

& et & ; %

Une liste plus complète des symboles utiles sous LATEX est disponible en

annexes, p. 262. Nous remarquons au passage que le backslash sert aussi de

caractère d’échappement pour tous les symboles utilisés lors de l’écriture

du code LATEX (&, $, #, \_, { ou } par exemple).

Les marges

Les gens me demandent souvent comment modifier les marges sous

LATEX. Personnellement, j’ai fini par m’habituer aux marges natives

de LATEX : je les modifie donc que pour des besoins très particuliers.

Si tu tiens à savoir pourquoi les marges sont plus grandes que celles

d’un document Word, c’est parce que LATEX a initialement été inventé

par des Américains (conventions américaines).

De plus, LATEX sert pour rédiger des rapports scientifiques : leur

reliure demande alors une marge plus importante s’ils sont épais.

Autrement, si tu as absolument besoin de modifier les marges

de ton document, je te recommande le package geometry, ainsi que

la page suivante : http://fr.wikibooks.org/wiki/LaTeX/Mise\_en\_

page#Modification\_des\_marges.

Bien, nous avons fait un premier tour d’horizon des premières possibi46

Chapitre 5. Gestion du texte et mise en forme

;A<

lités offertes par LATEX pour mettre en forme le texte. Passons à un peu

d’organisation.

5.3 Organiser son document

La base

Taper du texte brut, c’est bien. Mais mettre des titres, c’est mieux. Là

encore, rien à gérer. LATEX met en forme le titre selon son importance et

s’occupe de la numérotation. Par ordre d’importance, nous pouvons avoir :

→ \part{titre} ;

→ \chapter{titre} ;

→ \section{titre} ;

→ \subsection{titre} ;

→ \subsubsection{titre} ;

→ \paragraph{titre} ;

→ \subparagraph{titre}.

!

Nota Bene

La commande \part n’est disponible que pour un document de

classe report (cf. 4.3 La base d’un document LATEX p. 32).

De même, la commande \chapter n’est valable que pour les

classes book et report.

Si tu veux sauter une page, la commande \newpage est là. Un exemple

d’organisation serait donc :

Exemple d’organisation

% Enlever les % ici

%\part{Partie I}

%\section{Section 1.1}

Comment est-ce numéroté ?

47

Chapitre 5. Gestion du texte et mise en forme

;A<

%\section{Section 1.2}

D'une manière bizarre !

%\part{Partie II}

%\chapter{Chapitre 1}

%\section{Section 1.1}

Lorem ipsum\dots{}

\newpage

Bis repetita\dots{}

Configuration de la numérotation

Comme tu peux le constater, il y a quelques problèmes de numérotation.

Si les compteurs tournent normalement, il faut juste donner un coup de pouce

à LATEX pour faire correctement les choses. Retente le même code avec ces

commandes dans le préambule :

La numérotation des titres

% ATTENTION : écriture de ces commandes dans le PREAMBULE !!!

% RAZ des numéros de section après un chapitre

\makeatletter\@addtoreset{section}{chapter}\makeatother

% Pour mettre des I, II, etc. aux parties

\renewcommand{\thepart}{\Roman{part}}

% Pour mettre des 1, 2, etc. aux chapitres

\renewcommand{\thechapter}{\arabic{chapter}}

% Idem pour les sections et avoir le numéro de chapitre

\renewcommand{\thesection}{\thechapter.\arabic{section}}

Il existe plusieurs possibilités pour numéroter les différents titres de ton

48

Chapitre 5. Gestion du texte et mise en forme

;A<

document. La liste complète des commandes utilisables est la suivante :

❖ \arabic : pour avoir des chiffres arabes soit 1, 2, 3. . . ;

❖ \roman : pour avoir des chiffres romains minuscules soit i, ii, iii. . . ;

❖ \Roman : pour avoir des chiffres romains majuscules soit I, II, III. . . ;

❖ \alph : pour avoir des lettres minuscules soit a, b, c. . . ;

❖ \Alph : pour avoir des lettres majuscules soit A, B, C. . ..

Titre sans numérotation

Enfin, dans le cas où tu veux juste écrire un titre sans numéro, il suffit

d’ajouter une \* à la commande du titre : \part\*{titre}, etc.

Bon, c’est bien gentil d’avoir des titres. Comment obtenir un sommaire

désormais ?

5.4 Gestion du sommaire

La base

Pour une fois, nouvelle option, pas de nouveau package. Pour le sommaire,

tout est déjà inclus de base dans LATEX. Pour l’afficher, il faut juste renseigner

dans le code la commande \tableofcontents, à l’endroit où tu désires placer

le sommaire.

!

Un problème ?

« J’ai lancé la compilation du sommaire mais rien ne s’affiche

hormis Table des matières. Est-ce normal ? »

Oui. Dès lors que tu génères un sommaire, il faut toujours compiler deux fois pour obtenir le résultat final attendu.

À la première compilation, LATEX créé un nouveau fichier, d’extension .toc, où il stocke le sommaire. À la seconde, il regarde si un tel

fichier existe et, dans ce cas, récupère les informations pour générer

le sommaire.

49

Chapitre 5. Gestion du texte et mise en forme

;A<

!

Une question ?

« Je ne veux pas lire Table des matières mais Sommaire. Est-ce

possible ? »

Oui, tout à fait. C’est possible avec la commande suivante, dans

le corps du texte, juste avant \tableofcontents par exemple :

\renewcommand{\contentsname}{Sommaire}

Le nec plus ultra

Avoir un sommaire, c’est bien. Pouvoir interagir avec, c’est encore mieux !

Pour pouvoir se déplacer rapidement dans le document grâce à un clic sur

un titre du sommaire, il faut utiliser un nouveau package : hyperref.

Cependant, le résultat par défaut n’est pas très esthétique et peut entraîner une crise d’épilepsie aux plus sensibles, c’est pourquoi je recommande

d’utiliser l’option colorlinks 4

.

Mais hyperref va beaucoup plus loin et permet de personnaliser les options de lecture par défaut du PDF généré. C’est pourquoi je recommande

d’utiliser aussi les nombreuses options comme indiqué dans l’exemple ciaprès.

Ce même package permet d’indiquer des adresses Internet grâce à la commande \url{adresse\_internet}. En revanche, si certains liens sont trop

longs et finissent en fin de ligne, ils sortent de la page.

Ajouter breaklinks dans les options du package hyperref permet de

résoudre le problème. Des fois, charger le package supplémentaire url, avec

l’option hyphens, est obligatoire pour traiter les derniers cas de figure problématiques.

Le sommaire – Bilan

% Ajout au PREAMBULE

\usepackage[hyphens]{url} % Pour des césures correctes dans

les URLs

\usepackage[pdfauthor = {{Prénom Nom}}, pdftitle = {{Titre

document}}, pdfstartview = Fit, pdfpagelayout = SinglePage

4. Si tu ne me crois pas, génères un sommaire avec juste le package hyperref, sans

option, et regarde le rendu du fichier PDF. Tu comprendras.

50

Chapitre 5. Gestion du texte et mise en forme

;A<

, pdfnewwindow = true, bookmarksnumbered = true,

breaklinks, colorlinks, linkcolor = red, urlcolor = black,

citecolor = cyan, linktoc = all]{hyperref} % Renvois --

Options Adobe/lecteur PDF

\begin{document}

% Page de garde

% Sommaire -- Penser à compiler deux fois

{

\hypersetup{hidelinks} % Sommaire en "noir"

\renewcommand{\contentsname}{Sommaire} % Remplacer "Table des

matières"

\tableofcontents % Affichage du sommaire

}

% Si nécessaire

%\clearpage % Mieux qu'un \newpage ou des erreurs dans le

sommaire parfois

% Parties, chapitres, texte, etc.

% Commande fournie avec le package hyperref

\url{https://www.ctan.org/}

\end{document}

Pour revenir rapidement sur les options du package hyperref, en voici

un descriptif :

→ pdfauthor & pdftitle : pour renseigner correctement les champs des

options du fichier PDF.

Il est possible de remplir les autres champs disponibles : cf. la documentation du package hyperref 5

;

→ pdfstartview & pdfpagelayout : pour les options d’affichage du PDF

à sa lecture.

5. Et comme je suis adorable, voici le lien : https://www.ctan.org/pkg/hyperref.

51

Chapitre 5. Gestion du texte et mise en forme

;A<

Pour connaître toutes les options disponibles, cf. la documentation du

package hyperref ;

→ pdfnewwindow = true : si ton document contient un lien vers un autre

fichier PDF, cliquer sur le lien entraîne l’ouverture du PDF dans un

nouvel onglet (et non à la place du premier PDF) ;

→ bookmarksnumbered : pour les signets du lecteur PDF ;

→ breaklinks : pour permettre la césure des liens insérés trop longs ;

→ colorlinks et toutes les couleurs qui suivent : pour colorer correctement les références du document ;

→ linktoc = all : pour faire un renvoi du sommaire avec les numéros

de page.

Ajout d’un titre étoilé

Enfin, dans le cas des titres étoilés, ces derniers n’apparaissent pas dans

le sommaire. Il existe malgré tout un moyen de l’ajouter manuellement, si tu

y tiens. Cette solution requiert l’utilisation du package hyperref, que nous

connaissons déjà.

Ajout d’un titre étoilé dans le sommaire

% Ajout dans le préambule

%\usepackage{hyperref}

% Ajout d'un titre sans numéro

% Penser à enlever le % la ligne en dessous

%\section\*{Introduction} % Les titres doivent correspondre

\phantomsection % Renvoi correct dans le sommaire

\addcontentsline{toc}{section}{Introduction} % Ajout dans le

sommaire

Lorem ipsum dolor\dots{}

52

Chapitre 5. Gestion du texte et mise en forme

;A<

La ligne avec la commande \addcontentsline... sert à implémenter

dans le fichier .toc (fichier qui gère le sommaire) le titre Introduction en

tant que section (part si partie, chapter si chapitre, etc.).

Le numéro de page correspond à l’endroit où est tapée la commande, d’où

son positionnement après \section\*, pour éviter une mauvaise numérotation si le titre étoilé débute sur une nouvelle page.

Il faut donc faire bien attention avec cette situation pour garantir la cohérence du document : il faut renseigner le même titre dans \section\* et dans

\addcontentsline.... C’est pourquoi, personnellement, j’utilise les titres

étoilés le moins possible.

Bon, maintenant que nous avons un magnifique sommaire, est-il possible

d’ajouter une page de garde ?

5.5 La page de garde

La base

Comme pour le sommaire, il faut d’abord créer la page de garde puis

indiquer à LATEX de l’afficher. Pour la créer, il faut remplir les informations

suivantes :

La page de garde – Création

% Ajout dans le préambule ou après \begin{document}

% Titre

\title{Titre}

%\title{\textbf{Titre}} % Ressort mieux selon moi

% Auteur

\author{Prénom \textsc{Nom} \\ Profession}

% Date

\date{\today} % Date du jour (compilation du document)

%\date{date\_à\_afficher} % Date fixe

Naturellement, tout n’a pas besoin d’être renseigné. Si tu veux uniquement le titre, tu laisses juste la commande \title{Titre}.

53

Chapitre 5. Gestion du texte et mise en forme

;A<

La petite astuce

S’il y a plusieurs auteurs dans ton document, tu peux tous les

indiquer. Il faut juste les séparer par un \and, ce qui donne :

\author{Nom1 \and Nom2 \and Nom3}

LATEX s’occupe ensuite de la mise en forme de tous ces noms. Pratique, n’est-ce pas ?

Pour afficher la page de garde, il faut ensuite renseigner dans le corps du

document la commande \maketitle, de préférence dès le début.

Mais, tu devrais te rendre compte, après compilation, que, si tu demandes

à ton lecteur de fichier PDF d’aller à la page N, tu te retrouves en page N +1.

C’est parce que LATEX ne numérote pas la page de garde et commence ensuite

la numérotation à 1 au lieu de 2. . .

C’est peut-être un détail mais, personnellement, je trouve extrêmement

irritant d’aller dans le sommaire, de trouver le numéro de page de la section

qui t’intéresse, de la saisir dans ton lecteur PDF. . . et d’arriver à la mauvaise

page !

Fort heureusement, il suffit d’ajouter après \maketitle la commande

\setcounter{page}{2}, pour réajuster correctement la numérotation des

pages.

S’il fallait synthétiser les différentes options de base pour la page de garde,

nous pourrions alors nous servir du code suivant :

La page de garde – Bilan

\documentclass[a4paper, 12pt]{report}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage[french]{babel}

\usepackage{lmodern}

\begin{document}

% Informations de la page de garde

\title{\textbf{Initiation à \LaTeX{}} \\ \textit{Pour dé

54

Chapitre 5. Gestion du texte et mise en forme

;A<

butants ou jeunes utilisateurs}}

\author{Adrien \textsc{Bouzigues} \\ Salarié \and John \textsc

{Doe} \\ Profession}

\date{\today}

% Générer la page de garde

\maketitle

% Changer le titre du résumé

%\renewcommand{\abstractname}{\Large{}\textbf{Résumé}}

% Résumé

\begin{abstract}

Résumé du document

\end{abstract}

% Classe report : sur une page à part

% Classe article : sur la page de garde (si pas de newpage)

\clearpage\setcounter{page}{2}

Lorem ipsum dolor\dots{}

\end{document}

!

Une question ?

« Ce n’est pas pratique ta page de garde. C’est sobre, impossible

de mettre une image ! Est-il possible d’avoir mieux »

. . . Tu as parfaitement raison ! Il est tout à fait possible d’avoir

une page de garde plus personnelle et plus décorée. Pour satisfaire ta

curiosité, je te propose une première solution “simple” à utiliser.

Mais prends garde ! Dès l’instant où tu commences à arpenter

ce chemin – construire une page spécifique à partir de rien –, tu peux

très vite y passer beaucoup de temps. LATEX n’est pas un outil de

création graphique à la base.

Pour un rapport officiel ou si le temps t’est précieux, je recom55

Chapitre 5. Gestion du texte et mise en forme

;A<

!

mande d’utiliser les commandes de base que je viens de présenter.

Autrement, tu peux te permettre, comme je le fais pour ce guide,

de construire ta propre page personnalisée. Il n’y a pas une seule

bonne façon de faire et tout dépend de ce que tu veux faire.

Personnalisation de la page de garde

Comme promis, voici un exemple “simple” pour avoir une première page

de garde personnalisable. Les possibilités sont très nombreuses avec LATEX :

tout dépend donc de ce que tu veux faire.

Une solution personnalisable

\documentclass[a4paper, 12pt]{report}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage[french]{babel}

\usepackage{lmodern}

\usepackage{graphicx} % Pour insérer une image (logo)

% cf. les chapitres suivants pour plus de détails

\begin{document}

% Auteur : WikiBooks (http://en.wikibooks.org/wiki/LaTeX/Title

\_Creation)

% License : CC BY-NC-SA 3.0 (http://creativecommons.org/

licenses/by-nc-sa/3.0/)

% Adaptation du document d'origine

% Environnement titlepage : permet de créer une page de garde

et de la personnaliser à volonté

\begin{titlepage}

\newcommand{\HRule}{\rule{\linewidth}{0.5mm}} % Ligne

horizontale (épaisseur modifiable)

\begin{center} % Centrer le contenu de la page

% En-têtes

\textsc{\LARGE{}Université} \\[1.5cm]

56

Chapitre 5. Gestion du texte et mise en forme

;A<

\textsc{\Large{}En-tête principal} \\[0.5cm] % Nom du cursus (

par exemple)

\textsc{\large{}En-tête secondaire} \\[0.5cm] % Intitulé du

cours (par exemple)

% Titre

\HRule \\[0.6cm]

{\huge\bfseries{}Titre} \\[0.25cm]

\HRule \\[1.5cm]

% Auteur

\begin{minipage}{0.45\linewidth}

\begin{flushleft}

\Large\textit{Auteur :} \\

John \textsc{Smith} % Nom auteur

\end{flushleft}

\end{minipage}

\hfill

\begin{minipage}{0.45\linewidth}

\begin{flushright}

\Large\textit{Superviseur :} \\

Dr. John \textsc{Smith} % Nom superviseur

\end{flushright}

\end{minipage} \\[2cm]

% Si aucun superviseur, utiliser les lignes ci-après et

supprimer les lignes précédentes

%\Large\textit{Auteur :} \\

%John \textsc{Smith} \\[3cm] % Nom auteur

% Date

{\large\today} \\[2cm] % Date : \today ou date saisie à la

main

% Logo

%\includegraphics{logo.png} \\[1cm] % Logo à utiliser

\end{center}

\vfill % Remplir le reste de la page avec du blanc

\end{titlepage}

57

Chapitre 5. Gestion du texte et mise en forme

;A<

\clearpage\setcounter{page}{2}

Lorem ipsum dolor\dots{}

\end{document}

Allez, faisons une petite pause sur la mise en forme pour étudier un point

un peu abstrait mais extrêmement puissant et nécessaire pour poursuivre.

5.6 Création de commandes

Il peut arriver que tu aies besoin de cumuler des commandes, et ce, un

très grand nombre de fois. LATEX t’offre pour cela la possibilité d’en créer de

nouvelles.

Pour ce faire, il suffit d’ajouter la ligne suivante, de préférence dans le

préambule même si tu peux l’insérer n’importe où dans ton document (avant

le premier appel de ta nouvelle commande) :

\newcommand{nom\_commande}[nombre\_arguments]{commande}

Étudions son fonctionnement avec un exemple. Disons que je veuille

mettre un mot (ou un groupe de mots) en gras et en italique. Je vais donc

procéder ainsi :

Une première commande

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\newcommand{\grasital}[1]{\textbf{\textit{#1}}}

% Le nom de la commande commence par "\"

% La position de l'argument se fait avec un "#" et son numéro

\begin{document}

58

Chapitre 5. Gestion du texte et mise en forme

;A<

J'aime le chocolat ! \\

\grasital{J'aime le chocolat !} \\

J'aime le \grasital{chocolat} !

\end{document}

Note bien qu’il peut y avoir aucun argument comme plusieurs, avec une

limite de 9. Si l’utilisation avec plusieurs arguments sera plus concrète lorsque

nous aborderons les mathématiques, voici un cas sans argument :

Un second cas

% Création de la commande après

\begin{document} : OK

\newcommand{\SAV}{\textbf{Service

Après-Vente}}

Notre \SAV vous aide. \\

Grâce à notre \SAV, vous serez

comblés.

Notre Service AprèsVentevous aide.

Grâce à notre Service

Après-Vente, vous serez comblés.

J’ai décidé de mettre le texte en gras, mais rien ne m’empêche en cours

de rédaction de mon rapport de modifier ce choix. L’avantage ? Tu as juste à

modifier la commande et, lors de la compilation, la modification se répercute

sur tout le document ! Pratique, non ?

C’est aussi pourquoi je recommande de placer tous les newcommand

dans le préambule : c’est plus pratique pour les retrouver s’ils sont tous

au même endroit, au lieu d’être dispersés dans le document.

!

Une question ?

« Pourquoi, dans le second cas, n’y a-t-il pas d’espaces dans le

résultat entre SAV et propose ? »

Tu as l’œil ! Suite à une commande, LATEX ignore les espaces. Tu

peux en ajouter un à la fin de la commande mais il y en aura alors

aussi un après la virgule.

59

Chapitre 5. Gestion du texte et mise en forme

;A<

!

Pour indiquer à LATEX la fin de la commande, il faut donc la

fermer avec des accolades. C’est ce que je fais par exemple lorsque

j’écris LATEX : le code derrière est \LaTeX{}.

Il aurait donc fallu écrire dans mon exemple : notre \SAV{} vous

et notre \SAV{}, vous.

Tu ne trouves pas cet aspect utile pour l’instant mais tu verras que, quand

tu prendras un peu d’expérience, tu finiras par créer toi-même tes commandes

pour plus de simplicité et de rapidité.

Bien, continuons. C’est quoi déjà la suite ? Ah oui, les listes.

5.7 Les listes

La base

Les listes (à puces ou numérotées) sont très pratiques quand il s’agit

d’énumérer des éléments, faire un inventaire ou décrire des étapes.

Les listes peuvent donc être soit non numérotées (listes dites « à puces » :

tiret, rond, autres symboles), avec l’environnement itemize ; soit numérotées

(numéro ou lettre), grâce à l’environnement enumerate.

Pour définir une puce ou un numéro, il faut utiliser la commande \item.

Voyons plutôt le résultat avec un exemple :

Une liste à puces

Détail de la chambre :

\begin{itemize}

\item un lit ;

% Saut de ligne optionnel (aérer)

\item une armoire ;

\item et un bureau. \\

% Saut de ligne (\\) licite pour

aérer le texte

\end{itemize}

Détail de la chambre :

– un lit ;

– une armoire ;

– et un bureau.

60

Chapitre 5. Gestion du texte et mise en forme

;A<

Une liste numérotées

Pour écrire sous \LaTeX{}, il

faut :

\begin{enumerate}

\item Apprendre les bases.

\item Pratiquer les bases.

\item \^Etre curieux !

\end{enumerate}

% Imbrication des environnements

(et donc des listes) possible

Pour écrire sous LATEX, il

faut :

1) Apprendre les

bases.

2) Pratiquer les bases.

3) Être curieux !

Normalement, tu devrais avoir un résultat un peu différent du mieux : les

tirets sont plus grands et tes numéros se terminent par un point et non par

une parenthèse.

Tout va bien ! J’ai juste une configuration par défaut de LATEX qui génère

ce rendu. Tu peux donc constater que créer des listes sous LATEX se révèle

très facile. Voyons maintenant comment les personnaliser à notre guise ?

Personnalisation des listes

Pour personnaliser ses listes, il existe un package incontournable : le package enumitem. Combiné avec le package pifont pour obtenir des symboles

supplémentaires et donc avoir de nouvelles puces, c’est la meilleure combinaison possible pour personnaliser simplement ses listes.

Concrètement, le package enumitem offre quelques options, dont les suivantes que je recommande particulièrement :

→ label = puce : pour changer la puce ou la numérotation utilisée ;

→ leftmargin = \* : pour supprimer l’indentation de la liste ;

→ itemsep = <distance> : pour insérer <distance> entre 2 puces. C’est

plus commode ainsi que de devoir sauter des lignes avec \\[<distance>]

à chaque fin de puce.

Pour ce qui est du package pifont, son utilisation est très simple. Il faut

utiliser la commande \ding{<num>} avec <num> pris dans la liste ci-après

pour afficher un symbole du package.

61

Chapitre 5. Gestion du texte et mise en forme

;A<

Table 5.2 – Liste des symboles du package pifont

32 33 ✁ 34 ✂ 35 ✃ 36 ✄ 37 ☎ 38 ✆ 39 ✇

40 ✈ 41 ✉ 42 ☛ 43 ☞ 44 ✌ 45 ✍ 46 ✎ 47 ✏

48 ✐ 49 ✑ 50 ✒ 51 ✓ 52 ✔ 53 ✕ 54 ✖ 55 ✗

56 ✘ 57 ✙ 58 ✚ 59 ✛ 60 ✜ 61 ✝ 62 ✞ 63 ✟

64 ✠ 65 ✡ 66 ✢ 67 ✣ 68 ✤ 69 ✥ 70 ✦ 71 ✧

72 ★ 73 ✩ 74 ✪ 75 ✫ 76 ✬ 77 ✭ 78 ✮ 79 ✯

80 ✰ 81 ✱ 82 ✲ 83 ✳ 84 ✴ 85 ✵ 86 ✶ 87 ✷

88 ✸ 89 ✹ 90 ✺ 91 ✻ 92 ✼ 93 ✽ 94 ✾ 95 ✿

96 ❀ 97 ❁ 98 ❂ 99 ❃ 100 ❄ 101 ❅ 102 ❆ 103 ❇

104 ❈ 105 ❉ 106 ❊ 107 ❋ 108 ● 109 ❍ 110 ■ 111 ❏

112 ❐ 113 ❑ 114 ❒ 115 ▲ 116 ▼ 117 ◆ 118 ❖ 119 ◗

120 ❘ 121 ❙ 122 ❚ 123 ❛ 124 ❜ 125 ❝ 126 ❞

161 ❡ 162 ❢ 163 ❣ 164 ❤ 165 ❥ 166 ❦ 167 ❧

168 ♣ 169 ♦ 170 ♥ 171 ♠ 172 ① 173 ② 174 ③ 175 ④

176 ⑤ 177 ⑥ 178 ⑦ 179 ⑧ 180 ⑨ 181 ⑩ 182 ❶ 183 ❷

184 ❸ 185 ❹ 186 ❺ 187 ❻ 188 ❼ 189 ❽ 190 ❾ 191 ❿

192 ➀ 193 ➁ 194 ➂ 195 ➃ 196 ➄ 197 ➅ 198 ➆ 199 ➇

200 ➈ 201 ➉ 202 ➊ 203 ➋ 204 ➌ 205 ➍ 206 ➎ 207 ➏

208 ➐ 209 ➑ 210 ➒ 211 ➓ 212 ➔ 213 → 214 ↔ 215 ↕

216 ➘ 217 ➙ 218 ➚ 219 ➛ 220 ➜ 221 ➝ 222 ➞ 223 ➟

224 ➠ 225 ➡ 226 ➢ 227 ➣ 228 ➤ 229 ➥ 230 ➦ 231 ➧

232 ➨ 233 ➩ 234 ➪ 235 ➫ 236 ➬ 237 ➭ 238 ➮ 239 ➯

241 ➱ 242 ➲ 243 ➳ 244 ➴ 245 ➵ 246 ➶ 247 ➷

248 ➸ 249 ➹ 250 ➺ 251 ➻ 252 ➼ 253 ➽ 254 ➾

Mais le plus appréciable avec ce package enumitem, c’est la possibilité de

configurer globalement les liste dès le préambule (\setlist), voire d’en créer

de nouvelles avec ces propres règles, puces, distances, etc. (\newlist).

Voici un rapide aperçu des possibilités désormais accessibles en quelques

touches de clavier :

62

Chapitre 5. Gestion du texte et mise en forme

;A<

Des listes proprement personnalisables

% Ajout dans le préambule

%\usepackage{enumitem, pifont}

%\setlist[itemize, 1]{label =

{--}, itemsep = \baselineskip

}

%\setlist[enumerate, 1]{label =

\arabic\*), itemsep =

\baselineskip}

J'ai envie de dire : \begin{

itemize}

\item une chose ;

\item[\ding{118}] avec une puce

ponctuelle ! \\

\end{itemize}

Je peux aussi énumérer : \begin{

enumerate}[label = {\bfseries

{}\'Etape \Alph\* :},

leftmargin = \*]

\item marcher ;

\item lire ;

\item écrire.

\end{enumerate}

J’ai envie de dire :

– une chose ;

❖ avec une puce

ponctuelle !

Je peux aussi énumérer :

Étape A : marcher ;

Étape B : lire ;

Étape C : écrire.

OK pour toi ? Toujours d’attaque ? Finissons désormais ce chapitre sur la

gestion du texte.

5.8 Une petite touche de couleur ?

Nous avons vu beaucoup d’éléments de mise en page et de mise en forme

du texte mais tu conviendras qu’avoir un gros pavé en noir peut parfois être

rebutant à la lecture.

Pour mettre un peu de couleur, il faut d’abord charger le package xcolor

puis utiliser la commande :

63

Chapitre 5. Gestion du texte et mise en forme

;A<

\textcolor{nom\_couleur}{texte}

Les couleurs de base disponibles pour nom\_couleur sont alors les suivantes :

→ red ;

→ green ;

→ blue ;

→ cyan ;

→ magenta ;

→ yellow ;

→ orange ;

→ violet ;

→ purple ;

→ brown ;

→ black ;

→ darkgray ;

→ gray ;

→ lightgray ;

→ white .

Si jamais tu trouves qu’il n’y a pas assez de couleurs, tu peux utiliser

l’option dvipsnames dans le package (\usepackage[dvipsnames]{xcolor})

puis te référer à la Figure 5.1 pour nom\_couleur :

Figure 5.1 – Les couleurs avec dvipsnames

Enfin, si jamais tu trouves que tu n’as toujours pas assez de couleur pour

laisser ton talent artistique s’exprimer, sache qu’il est possible d’en créer dans

le préambule avec la commande :

\definecolor{nom\_couleur}{modèle}{def\_couleur}

64

Chapitre 5. Gestion du texte et mise en forme

;A<

Le modèle correspond à RGB par exemple et def\_couleur à 255,215,0

(couleur or). Pour plus de renseignements quant à cette commande, tu peux

consulter la page suivante : http://fr.wikibooks.org/wiki/LaTeX/Options\_

de\_mise\_en\_forme\_avanc%C3%A9es#Mod.C3.A8les\_de\_couleur.

Tu peux aussi renseigner nom\_couleur ou def\_couleur par l’intermédiaire de mélanges de couleur. Pour ce faire, il faut utiliser la syntaxe

couleurA!x!couleurB, pour x ∈ [0 ; 100], qui te permet de mélanger x % de

couleurA et (100 − x) % de couleurB.

Sache qu’il existe aussi des commandes comme \colorbox ou \pagecolor.

Je te laisse aller te renseigner si tu es intéressé pour te laisser un peu en autonomie.

Cette fois, nous en avons fini avec le texte et sa mise en forme. Tout

d’abord, une référence s’impose :

Figure 5.2 – Non, je ne suis pas un fan d’Evangelion !

Toujours des nôtres ? Si tu te sens prêt, nous allons pouvoir aborder un

nouveau chapitre !

65

Chapitre 6

Les mathématiques sous LATEX

C

omme tu peux le constater, il y a fort à faire sous LATEX. Les combinaisons sont déjà impressionnantes. Je te laisse maintenant découvrir

la raison d’être de LATEX, ce pourquoi il a été créé : écrire proprement des

formules mathématiques !

Par la suite, pour alléger les exemples, le préambule ne sera plus renseigné

dans les codes LATEX mis à disposition. Ces derniers seront basés sur l’architecture du code minimal fourni ci-après. L’ajout de nouveaux packages sera

signalé au début du code par un commentaire.

Le code minimal

\documentclass[a4paper, 12pt]{report}

% PDFLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\begin{document}

% Ecrire le code ici !

\end{document}

66

Chapitre 6. Les mathématiques sous LATEX

;A<

6.1 Le mode mathématiques

Bon, c’est bien beau de vouloir faire écrire des maths à LATEX, encore fautil lui indiquer qu’il s’agit justement de maths ! C’est le principe du « mode

mathématiques ».

Celui-ci est défini soit par le symbole $ (un ouvrant et un fermant),

soit par des “backslash-crochets” \[ et \], soit par l’environnement equation :

Le mode mathématiques

\'Ecrire x = 2 ! et $x = 2 !$ ne

donnent pas le même résultat

! \\

De même si j'écris \[x = 2 !\] %

Pas de \\ car déjà un saut de

ligne

On obtient la même chose avec :

\begin{equation}

x = 2 !

\end{equation}

mais l'équation est numérotée !

\\

Que donne $a b c d$ ?

Écrire x = 2 ! et x = 2!

ne donnent pas le même

résultat !

De même si j’écris

x = 2!

On obtient la même

chose avec :

x = 2! (6.1)

mais l’équation est numérotée !

Que donne abcd ?

Repérer le mode mathématiques sous Texmaker

Il est très aisé de voir si du texte est en mode mathématiques :

Texmaker affiche ce texte en vert !

Le conseil personnel

J’utilise très peu l’environnement equation, sauf quand j’ai besoin de numéroter des formules. Si je n’ai pas besoin de numérotation,

l’environnement equation\* donne le même résultat que \[ et \].

67

Chapitre 6. Les mathématiques sous LATEX

;A<

De ce que tu as pu observer dans l’exemple, le mode mathématiques met

le texte en italique et supprime les espaces. En effet, dans ce mode, LATEX

considère que tout ce qui est écrit n’est que produit (comme pour l’exemple

$a b c d$).

L’utilisation de \[ et \] permet d’aller à la ligne et de centrer la formule.

Cette option est très pratique pour présenter un résultat ou une longue équation.

!

Une question ?

« Si le mode mathématiques revient à mettre du texte en italique,

pourquoi ne pas écrire du texte et utiliser la commande \textit ? »

Outre l’aspect esthétique de la formule, le mode mathématiques

est le seul mode qui tolère et permette d’appeler les commandes que

nous verrons par la suite, pour écrire des formules mathématiques

(fraction, somme, intégrale, dérivée partielle. . .).

S’il est possible de mélanger le mode mathématiques avec du texte grâce

au $, c’est plus délicat avec les autres commandes, tant pour l’écriture en

italique que pour l’absence d’espace. Mais il existe une solution.

6.2 Vers les espaces insécables

Pour pouvoir librement écrire du texte dans le mode mathématiques, la

commande \text{texte\_à\_écrire} est très utile. C’est vraiment la commande la plus simple qui existe : à utiliser en priorité pour ce genre

de situation !

Mais, tu peux aussi avoir envie de jouer un peu sur l’espacement entre les

différents symboles, si tu trouves qu’ils sont trop rapprochés. Il existe alors

des commandes bien plus efficaces et pratiques que \hspace{<distance>}

pour le mode mathématiques.

Ces commandes portent le nom d’espaces insécables – insécables car

LATEX ne peut y toucher et se plie à la volonté de l’utilisateur. Ces espaces

sont utilisables aussi bien dans le mode mathématiques que sur du

texte normal.

Ils permettent aussi de bien imposer l’espace souhaité et évite d’avoir un

symbole ou un signe de ponctuation qui se balade seul en début de phrase.

Nous pouvons relever :

68

Chapitre 6. Les mathématiques sous LATEX

;A<

❖ \! : espacetrèspetit,

❖ \, : espace fin,

❖ \: : espace moyen,

❖ ~ (tilde) : espace normal,

❖ \; : espace large,

❖ \quad : espace très large,

❖ \qquad : espace encore

plus large.

Toutefois, si j’utilisais initialement les espaces insécables à outrance, ils

peuvent vite se révéler pénibles à écrire. Il faut donc généralement faire

confiance à LATEX pour la mise en forme et les utiliser avec parcimonie.

Personnellement, je les utilise surtout, par exemple, après le symbole ∀

(\forall) car l’espacement est très faible. À toi de choisir ta préférence :

∀x vs ∀ x

$\forall x$ vs $\forall\,x$

Commande et espaces insécables

Nous obtenons donc $x + y = 3$ et

$y = 2$ donc : \[x = 1 \text

{ (obvious)}\]

% Présence d'un espace au début

dans \text : séparation du

texte de l'équation

% Utiliser \quad aussi possible :

exemple d'utilisation assez

fréquent

Nous obtenons alors : \[x = 1

\quad \text{et} \quad y = 2\]

Nous obtenons donc x +

y = 3 et y = 2 donc :

x = 1 (obvious)

Nous obtenons alors :

x = 1 et y = 2

Si jamais tu veux appliquer un espace insécable de manière définitive sur

une commande LATEX, il existe des moyens de remplacer la définition initiale

de la commande par la même avec l’espace insécable.

Ainsi, tu continuerais à écrire $\forall x$ mais le résultat serait identique à $\forall\,x$. Il faut procéder de la manière suivante dans le préambule :

69

Chapitre 6. Les mathématiques sous LATEX

;A<

Changement de la définition d’une commande

% Renommer la commande initiale (sinon bug : boucle infinie)

\let\oldforall\forall

% Modification de la commande

\renewcommand{\forall}{\oldforall\,}

Bref, après cette brève initiation aux mathématiques, allons vraiment

écrire des formules mathématiques.

6.3 Des exemples de formules

Avant de se lancer, les mathématiques n’échappent pas à la règle : il faut

charger des packages avant de commencer.

Après plusieurs recherches, je recommande amsmath, amsfonts et amssymb.

Il semblerait que ces trois packages suffisent pour traiter 95 % des formules

mathématiques. Commençons donc par un premier exemple :

Les premiers symboles mathématiques

% Ajout au préambule !

%\usepackage{amsmath, amsfonts,

amssymb}

Indice : $i\_2$ \\

% Encadrement avec des {}

$i\_{13}$ différent de $i\_13$ \\

Exposant : $i^3$ ou $i^{13}$ \\

Fraction : $\frac{x}{y}$ \\

Racine carrée : $\sqrt{13}$ \\

Racine énième : $\sqrt[n]{13}$ \\

Mix de formules (exemple) : $

\sqrt{\frac{a}{b}}$

Indice : i2

i13 différent de i13

Exposant : i

3

ou i

13

Fraction : x

y

Racine carrée : √

13

Racine énième : √n

13

Mix de formules

(exemple) : ra

b

Pardon ? Il n’y en a pas assez ? Ok, navré, poursuivons :

70

Chapitre 6. Les mathématiques sous LATEX

;A<

D’autres symboles mathématiques

Intégrale : $\int\_0^{13} f(x)\,dx

$ \\

% Attention aux bornes : les {}

sont vite oubliées

Somme : $\sum\_{i = 13}^n x^i$ \\

\'Equation : $x + y - z = 3

\times t + f$ \\

% Symbole +, - et = au clavier ;

\times pour un produit

$x < y$, $y \leq z$, $z \geqslant

c$, $c > d$ mais $d \neq f$

alors que $f \simeq g$ ! \\

% D'autres symboles - A toi de

voir si tu préfères \leq à

\leqslant (idem pour \geq)

Intégrale : Z 13

0

f(x) dx

Somme : Xn

i=13

x

i

Équation : x + y − z =

3 × t + f

x < y, y ≤ z, z ⩾ c,

c > d mais d ̸= f alors

que f ≃ g !

Ok pour toi ? Comment ? J’ai oublié de mentionner les lettres grecques ?

Toutes mes excuses. Les voici :

Les lettres grecques

Les lettres grecques ? Facile : $

\alpha$, $\beta$, $\mu$, etc.

\\

En majuscules ? $\Omega$, $\Delta

$, $\Lambda$, etc.

% Ne fonctionne pas pour toutes

les majuscules : \Alpha entra

îne une erreur

Les lettres grecques ?

Facile : α, β, µ, etc.

En majuscules ? Ω, ∆, Λ,

etc.

Si jamais tu souhaites connaître la liste exacte des commandes pour écrire

les lettres grecques, la voici :

71

Chapitre 6. Les mathématiques sous LATEX

;A<

Table 6.1 – La liste complète des lettres grecques sous LATEX

α \alpha η \eta ξ \xi τ \tau

β \beta θ \theta π \pi υ \upsilon

γ \gamma ϑ \vartheta ϖ \varpi ϕ \phi

δ \delta κ \kappa ρ \rho φ \varphi

ϵ \epsilon λ \lambda ϱ \varrho χ \chi

ε \varepsilon µ \mu σ \sigma ψ \psi

ζ \zeta ν \nu ς \varsigma ω \omega

Γ \Gamma Λ \Lambda Σ \Sigma Ψ \Psi

∆ \Delta Ξ \Xi Υ \Upsilon Ω \Omega

Θ \Theta Π \Pi Φ \Phi

Utiliser Texmaker

Que ce soit pour les lettres grecques ou plein d’autres éléments

mathématiques, Texmaker offre des raccourcis sur le côté gauche de la

fenêtre.

N’hésite pas à aller jeter un coup d’œil au début. Je trouve que

c’est mieux de taper les commandes mais il faut bien les avoir vues

une ou deux fois avant pour savoir qu’elles existent.

!

Une question ?

« J’ai tenté un mathrm sur une lettre grecque pour enlever son

“caractère italique” mais ça n’a pas fonctionné. . . »

Ah, j’ai affaire à un petit malin (qui a le mérite d’être allé fouiner

une nouvelle commande). Tout d’abord, \mathrm est une commande

qui ne fonctionne qu’en mode mathématiques et qui permet de redresser le texte (enlever l’italique). Il faut donc bien écrire :

$\mathrm{\mu}$

Cependant, mathrm ne fonctionne pas dans le cas des lettres

grecques. C’est pourquoi tu peux ajouter le package suivant :

72

Chapitre 6. Les mathématiques sous LATEX

;A<

!

upgreek. Il permet d’écrire les lettres grecques droites. La commande $\upmu$ est censée fonctionner.

Attention, ce package ne concerne pas toutes les lettres grecques !

\upOmega ne fonctionne pas car Ω est déjà considérée comme droite.

Cette commande est donc à manier avec prudence et qu’en cas de

nécessité absolue : les lettres grecques en italique rendent déjà très

bien.

Bon, je crois que nous avons déjà pas mal fait le tour. J’ai bâillonné l’élève

curieux qui voulait savoir comment améliorer l’affichage de la fraction, de la

somme et de l’intégrale : nous allons traiter ce point immédiatement.

6.4 L’affichage et les délimiteurs

Tu l’as peut-être remarqué : écrire une somme doit donner un résultat un

peu différent de ce que tu peux lire sur cette page. Il doit en aller de même

si tu écris une intégrale ou un empilement de fraction :

X

k

x

k

Z x

0

(f t) dt

a

b

c

d

Pour avoir un affichage “normal”, il faut indiquer à LATEX de forcer toutes

les équations en mode mathématiques avec l’affichage displaystyle.

La commande \everymath{\displaystyle} juste après \begin{document}

suffit donc pour avoir le même rendu que moi. . . sauf pour les fractions. Pour

ces dernières, il faut utiliser la commande \cfrac{x}{y} ou \dfrac{a}{b}

Dès lors, l’affichage de tes équations devrait être meilleur :

Forcer l’affichage

% Toujours dans le préambule

%\usepackage{amsmath, amsfonts, amssymb}

\everymath{\displaystyle} % Commande indispensable !

Somme : $\sum\_k x^k$ \\

73

Chapitre 6. Les mathématiques sous LATEX

;A<

Intégrale : $\int\_0^x (ft)\,dt$ \\

Fractions : $\frac{\frac{a}{b}}{\frac{c}{d}} \neq

\cfrac{\cfrac{a}{b}}{\cfrac{c}{d}}$

Bien, maintenant que les choses sont correctement posées, tu peux avoir

le meilleur rendu au monde mais LATEX reste toujours extrêmement puissant,

à condition de le lui dire.

En effet, écrire (

a

b

) et

a

b

!

sont deux choses totalement différentes. LATEX

est donc capable d’adapter la taille des parenthèses, crochets, accolades et

autres, en mode mathématiques, et toujours à condition de le lui signaler.

Cette particularité est appelé un délimiteur.

Les règles élémentaires des délimiteurs

Règle no 1 : Un délimiteur n’existe qu’en mode mathématiques.

Règle no 2 : Un délimiteur entrant implique un délimiteur

sortant.

Règle no 3 : Un délimiteur entrant est défini par la commande \left

suivi du nom du délimiteur ; pour le sortant, de même

avec \right.

Règle no 4 : Si tu ne veux pas afficher un délimiteur, il faut utiliser

la commande \left. ou \right. (il y a un point à la

fin).

Voyons de suite les noms des délimiteurs et leur fonctionnement avec un

exemple :

74

Chapitre 6. Les mathématiques sous LATEX

;A<

Les délimiteurs en action

% Toujours dans le préambule

%\usepackage{amsmath, amsfonts,

amssymb}

Parenthèses : $\left( \cfrac{a}{b

} \right)$ \\

Crochets : $\left[ \cfrac{a}{b}

\right]$ \\

Mix possible : $\left( \cfrac{a}{

b} \right]$ \\

% Aucun problème tant que la rè

gle 2 est respectée

Accolade à gauche : $\left\{

\cfrac{a}{b} \right.$ \\

Accolade à droite : $\left.

\cfrac{a}{b} \right\}$ \\

Bonus $\left\langle \left\lbrace

\cfrac{a}{b} \right\rbrace

\right\rangle$ \\

% \lbrace ou \rbrace équivalent à

\{ ou \}

% Selon moi : \{ plus logique

Parenthèses :

a

b

!

Crochets : "

a

b

#

Mix possible :

a

b

#

Accolade à gauche : (

a

b

Accolade à droite :

a

b

)

Bonus \*(a

b

)+

Il faut aussi savoir que les délimiteurs sont parfois inutiles :

(a × b) vs (a × b)

$(a \times b)$ vs $\left( a \times b \right)$

Les délimiteurs sont donc pratiques et intéressants à utiliser dès lors qu’il

y a un “étage” dans l’équation. Autrement, mieux vaut les éviter, pour simplifier l’écriture des équations et réduire les erreurs.

C’est bon ? Pas de questions ? Ouah, je dois commencer à bien expliquer

les choses pour une fois ! La suite ? Une petite escale dans le monde des

matrices. . .

75

Chapitre 6. Les mathématiques sous LATEX

;A<

6.5 Les matrices

Je préfère le répéter au cas où mais il faut naturellement employer le

mode mathématiques. En revanche, pas besoin de nouveaux packages. Tout

est déjà inclus avec les trois de base (amsmath, amsfonts et amssymb).

Ce n’est pas compliqué mais c’est aussi soumis à quelques règles. Je préfère

donc bien les poser maintenant car nous en aurons besoin un peu plus loin

dans ce guide :

Règles de base pour les matrices – Introduction aux tableaux

Règle no 1 : Il faut considérer une matrice n × m comme un tableau

vide à n × m cases.

Règle no 2 : Une matrice est générée par l’environnement pmatrix.

Règle no 3 : Les colonnes sont séparées par une esperluette “&”

(touche 1 sous Windows).

Règle no 4 : Le passage à la ligne suivante se fait grâce à \\.

De même, ne nous privons pas d’un petit exemple pour comprendre et

digérer le tout :

Les matrices – 1ers exemples

% Toujours dans le préambule

%\usepackage{amsmath, amsfonts,

amssymb}

Matrice 2 x 2 : $\begin{pmatrix}

a & b \\ c & d

\end{pmatrix}$ \\

Matrice 2 x 4 : \[\begin{pmatrix}

a & b & c & d \\

e & f & g & h

\end{pmatrix}\]

Matrice 2 x 2 :

a b

c d

Matrice 2 x 4 :

a b c d

e f g h

76

Chapitre 6. Les mathématiques sous LATEX

;A<

Il restera toujours des cas un peu plus délicats à traiter :

Les matrices – Cas plus technique

% Toujours dans le préambule

%\usepackage{amsmath, amsfonts,

amssymb}

Matrice à trou : \[\begin{pmatrix

}

1 & 1 & \cdots & 1 \\

1 & 2 & \cdots & 2 \\

\vdots & \vdots & \ddots & \vdots

\\

1 & 2 & \cdots & n

\end{pmatrix}\]

Matrice à trou :

1 1 · · · 1

1 2 · · · 2

.

.

.

.

.

.

.

.

.

.

.

.

1 2 · · · n

Comme indiqué dans les règles, il faut donc utiliser le symbole & pour

changer de colonne et la commande \\ pour passer à la ligne suivante. Si

l’espace entre les & est optionnel, il est quand même recommandé pour faciliter la relecture de ton code.

Surtout au début, pour des matrices plus complexes comme des matrices

à trou, il ne faut pas hésiter à faire un dessin pour bien visualiser l’agencement des différents éléments de la matrice.

Il n’existe pas qu’un seul environnement pour écrire des matrices. Nous

pouvons relever :

❖ matrix : aucun délimiteurs ;

❖ pmatrix : parenthèses ;

❖ vmatrix : barres verticales ;

❖ Vmatrix : doubles barres verticales ;

❖ bmatrix : crochets ;

❖ Bmatrix : accolades.

Un bon exemple de création de commande avec plusieurs arguments intervient ici. J’ai eu un jour à rédiger un corrigé d’exercices de physique.

Ce corrigé contient énormément de vecteurs. J’ai donc inventé la commande

vcol de la manière suivante :

77

Chapitre 6. Les mathématiques sous LATEX

;A<

\newcommand{\vcol}[3]

{\begin{pmatrix} #1 \\ #2 \\ #3 \end{pmatrix}}

qui s’appelle de cette façon : $\vcol{a}{b}{c}$. Plus pratique, n’est-ce pas ?

Je te garantis que c’est vrai, surtout que tu dois écrire un très grand nombre

de fois un vecteur colonne sur la même page !

Bon, finissons-en avec les mathématiques sous LATEX par la présentation

et l’alignement des équations.

6.6 Aligner des équations

Comme une image sera plus parlante que des mots, j’aimerais obtenir ce

résultat :

« Nous cherchons a tel que :

P(µ ∈ I) = 1 − α

= 0, 9

car l’énoncé indique que 1 − α = 0, 9

= P

X¯ − µ

S

√

n − 1 ∈

"

−

a

S

√

n − 1 ;

a

S

√

n − 1

#!

= 2Sn−1

a

S

√

n − 1

!

− 1

Nous pouvons donc conclure par 1

:

I = [74, 98 − 0, 0428 ; 74, 98 + 0, 0428]

n = 20

1 − α = 0, 9

. »

Pour obtenir ce résultat avec des équations bien alignées, tu dois utiliser

l’environnement align (ou align\* pour éviter la numérotation de chaque

ligne).

1. Si c’est du chinois pour toi, je te rassure, ce sont des statistiques !

78

Chapitre 6. Les mathématiques sous LATEX

;A<

!

Ne pas utiliser eqnarray ! ! !

Après avoir vagabondé sur Internet et essayé différents rendus,

je préfère utiliser align. Après d’autres recherches, Lars Madsen

préconise lui aussi très fortement l’usage de align et recommande

de bannir toute utilisation de eqnarray (un autre environnement

pour aligner des équations) a

.

S’il y a donc un point à retenir : « Avoid eqnarray ! » et utilise

bien l’environnement align.

a. Article disponible sur : http://www.tug.org/pracjourn/2006-4/madsen/

madsen.pdf.

Pour le second résultat avec des accolades, il faut utiliser les délimiteurs

et un tableau avec l’environnement array.

Si array fonctionne en mode mathématiques, fais attention : align

s’emploie sans ! C’est parti pour un exemple. Reproduisons le cas présent en

page 78 :

Aligner des équations – 1ère partie

% Toujours dans le préambule

%\usepackage{amsmath, amsfonts,

amssymb}

Nous cherchons $a$ tel que :

\begin{align\*}

P\,(\mu \in I) & = 1 - \alpha \\

& = 0,9 \intertext{car l'é

noncé indique que $1 - \alpha

= 0,9$}

& = P \left( \cfrac{\bar{X} -

\mu}{S} \dots{} \right) \\

& = 2 \mathcal{S}\_{n - 1}

\left( \cfrac{a}{S} \dots{}

\right)

\end{align\*}

Nous cherchons a tel

que :

P (µ ∈ I) = 1 − α

= 0, 9

car l’énoncé indique que

1 − α = 0, 9

= P

X¯ − µ

S

. . . !

= 2Sn−1

a

S

. . . !

Que relevons-nous de concret sur ce premier cas de figure ?

79

Chapitre 6. Les mathématiques sous LATEX

;A<

❖ l’environnement align (ou align\*) utilise un & – et un seul – comme

point de repère pour l’alignement. C’est pourquoi il est plutôt recommandé de le placer avant le signe = ;

❖ une nouvelle ligne est annoncée par un saut de ligne \\, comme pour

une matrice ;

❖ la commande \intertext{texte} permet d’ajouter une remarque entre

2 équations. Notons au passage l’absence (volontaire) de saut de ligne

(\\) : \intertext entraîne déjà un espacement suffisant ;

❖ plus anecdotique : la commande \mathcal{texte} permet de “transformer” les caractères (utilisation d’une autre police adaptée aux symboles mathématiques).

Aligner des équations – 2nde partie

% Toujours dans le préambule

%\usepackage{amsmath, amsfonts,

amssymb}

Nous pouvons donc conclure par :

\[\left\{\begin{array}{rcl}

I & = & [74,98 \dots{} ] \\

n & = & 20 \\

1 - \alpha & = & 0,9

\end{array} \right.\]

Nous pouvons donc

conclure par :

I = [74, 98 . . . ]

n = 20

1 − α = 0, 9

Nous constatons que l’environnement array fonctionne de manière très

similaire aux matrices. Il faut indiquer le nombre de colonnes via des lettres

(l, c ou r).

Le nombre de lettres correspond au nombre de colonnes et le nom parle

de lui-même pour positionner le contenu à l’intérieur de la colonne : left,

center ou right.

L’utilisation d’un délimiteur est parfaitement envisageable (et recommandé) pour avoir l’accolade de taille variable à gauche. L’usage du mode

mathématiques devient dès lors obligatoire et justifie l’emploi de

l’environnement array au lieu d’align.

80

Chapitre 6. Les mathématiques sous LATEX

;A<

Enfin, si jamais tu désires avoir une résolution d’équations avec un seul numéro global comme référence (ce que ne permet pas l’environnement align),

tu peux procéder de la façon suivante :

Des équations – Un numéro

% Toujours dans le préambule

%\usepackage{amsmath, amsfonts,

amssymb}

\begin{equation}

\begin{split}

x & = y + z \\

& = 13

\end{split}

\end{equation}

x = y + z

= 13

(6.2)

Il faut donc utiliser l’environnement equation pour passer en mode mathématiques avec un numéro pour l’équation, puis utiliser l’environnement

split pour écrire tes équations bien alignées.

L’environnement split fonctionne de la même manière que l’environnement align (ou align\*).

Que vais-je bien pouvoir t’expliquer désormais ? Et surtout, comment

vais-je bien pouvoir remplir le bas de cette page avant de passer au prochain

chapitre. . .

Tu auras le droit à un gâteau si tu

me croises un jour et que tu me

donnes l’origine de cette image. Et

il y a une référence dans cette référence. . . #The cake is a lie !

81

Chapitre 7

Les tableaux et boîtes sous

LATEX

S

ynthétiser l’information n’est pas toujours évident. Et pourtant, un

bon tableau suffit parfois à véhiculer un message ou à lister des éléments. Voyons comment en créer sous LATEX.

Par la suite, pour alléger les exemples, le préambule ne sera plus renseigné

dans les codes LATEX mis à disposition. Ces derniers seront basés sur l’architecture du code minimal fourni ci-après. L’ajout de nouveaux packages sera

signalé au début du code par un commentaire.

Le code minimal

\documentclass[a4paper, 12pt]{report}

% PDFLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\begin{document}

% Ecrire le code ici !

\end{document}

82

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

7.1 Conventions

C’est le passage un peu rébarbatif du guide mais qu’il faut rappeler si tu

t’intéresses un peu à la mise en forme de documents. Les conventions que je

vais énoncer ci-après proviennent à l’origine du guide du package booktabs 1

,

qui permet d’améliorer la qualité des tableaux sous LATEX.

Les conventions pour rédiger des tableaux

«

Tu ne feras pas de graves erreurs si tu te rappelles à tout moment

de deux simples commandements :

1) Ne jamais, au grand jamais, utiliser de filets verticaux.

2) Ne jamais utiliser de filets doubles.

Ces commandements peuvent sembler extrêmes mais en des années

d’expérience je n’ai jamais trouvé un bon argument pour passer outre.

Par exemple, si tu estimes que les informations dans la moitié

gauche d’une table sont si différentes de celles de la droite qu’il faut

les séparer par une ligne verticale, alors tu devrais plutôt utiliser deux

tables.

Le second commandement est très, très occasionnellement violé.

[. . .]

Il y a trois autres conseils que je pourrai citer ici car ils sont si peu

connus en dehors des cercles des typographes et éditeurs professionnels :

1) Place les unités dans l’en-tête de la colonne (pas dans le corps

de la table).

2) Fais toujours précéder un point décimal (une virgule décimale

en français) par un chiffre ; donc 0.1 (ou 0,1) et pas simplement

.1 (,1).

3) N’utilises pas de signes « ditto » ou toute convention analogue

pour répéter une valeur précédente. Dans la plupart des cas, un

1. Disponible sur : https://www.ctan.org/pkg/booktabs.

83

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

blanc fait aussi bien l’affaire. Sinon, répètes la valeur.

Est-ce que c’est moi qui suis pédant ? Ces derniers conseils sont de

plus en plus souvent ignorés dans les travaux publiés. Pour moi, ceci

montre simplement que la typographie est celle d’un amateur.

»

Guide du package booktabs

https://www.ctan.org/pkg/booktabs

Maintenant que les conventions sont posées, voyons désormais comment

créer un tableau.

7.2 Création de tableaux

Nativement, tous les éléments sont disponibles sous LATEX pour créer des

tableaux extrêmement simples. Pour ce faire, il faut utiliser l’environnement

tabular.

Si tu te souviens bien de la construction de l’environnement array (tableaux en mode mathématiques), tu vas vite te rendre compte que le fonctionnement de base de l’environnement tabular (tableaux en mode texte)

est identique. Un petit exemple, comme toujours :

Premiers tableaux

\begin{tabular}{cc}

Centrage & Ici aussi \\

Ok ! & $\alpha = 13$

\end{tabular} \\ \\

\begin{tabular}{ll} \hline

Tableau & simple \\ \hline

sous & \LaTeX{} \\ \hline

\end{tabular}

Centrage Ici aussi

Ok ! α = 13

Tableau simple

sous LATEX

Comme tu peux le constater, je ne t’ai pas menti : la construction est

rigoureusement identique à celle de l’environnement array. La seule différence ? Comme il s’agit d’un tableau en mode texte, il est tout à fait licite

84

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

d’introduire un mode mathématiques local (avec des $) pour écrire des mathématiques dans une cellule du tableau.

Pour la séparation avec un filet 2

horizontal, il faut donc appeler la commande \hline après un saut de ligne (hormis au début du tableau).

!

Une question ?

« Par rapport à ton exemple, mes tableaux sont plus resserrés.

Pourquoi n’avons-nous pas le même résultat ? »

Question très pertinente : j’ai en effet une corde supplémentaire à

mon arc. J’ai indiqué dans le préambule de ce guide une commande

qui impacte tous mes tableaux et permet de les aérer un peu plus.

Cette commande est la suivante :

\renewcommand{\arraystretch}{1.3}

Elle permet d’agrandir la hauteur minimale d’une ligne, ce qui

permet d’aérer les tableaux. Le coefficient de 1.3 est un choix personnel : libre à toi de le modifier à ta convenance.

La petite astuce

Si jamais tu as un “grand nombre” de colonnes à déclarer lors de

la création de ton tableau, il existe un petit raccourci.

Au lieu d’écrire c...c (N fois), tu peux écrire \*{N}{|c}|. Ainsi, tu

crées N colonnes centrées. Pratique, non ?

Si tu veux des options plus poussées sur les tableaux (fusion de cellules,

remplissage, mise en gras d’une colonne entière. . .), je te renvoie à la 3ème

partie de ce guide où tu peux trouver des réponses. Internet peut aussi t’aider

si besoin.

Sache encore que, dès l’instant où tu arpentes le chemin d’une personnalisation très poussée et sophistiquée, tu risques de perdre beaucoup de temps

à faire en sorte que le code LATEX fonctionne. Avec les éléments de base que

je viens de te présenter, j’estime que tu peux déjà faire 70 % des tableaux

nécessaires.

Pour les 30 % restants, à titre indicatif et si tu es curieux, tu peux te tourner vers les packages suivants : array, booktabs, longtable et multirow

2. Terme consacré apparemment : c’est l’équivalent d’un “trait”.

85

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

(fusion de lignes ; fusion de colonnes possible de base avec la commande

\multicolumn).

Maintenant que les éléments de base ont été présentés, passons à un autre

élément important.

7.3 Insérer une légende

Avoir un tableau, c’est bien. Avoir un tableau avec une légende, c’est

mieux. Et avoir une légende avec une numérotation automatique, c’est encore

mieux ! Fort heureusement, LATEX propose tous ces éléments nativement.

Je ne vais pas rentrer dans les détails du concept, que je développe plus

amplement dans le prochain chapitre. Sache juste, pour commencer, qu’il te

faut procéder de la manière suivante :

1) Insertion du tableau dans un environnement table.

2) Insertion de la légende avant ou après le tableau (au choix) grâce à la

commande \caption{Légende}.

Un exemple minimaliste serait alors le suivant :

Tableau & légende

\begin{table}

\centering

\caption{Légende du tableau}

\begin{tabular}{ccc}

Tableau & de & test \\ \hline

sous & \LaTeX{}

\end{tabular}

\end{table}

Passons maintenant à un autre élément disponible sous LATEX pour faire

un peu de mise en forme sans nécessairement passer par un tableau : les

boîtes.

86

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

7.4 Les boîtes

La théorie

Si tu peux tout à fait utiliser un tableau pour encadrer des formules, des

images ou du texte, il existe d’autres solutions plutôt complètes et personnalisables sous LATEX. En l’occurrence, parlons des boîtes.

Sous LATEX, tout tient dans une boîte : les lettres, les paragraphes, les tableaux, les images, les équations. . . Bref, tout ! Concrètement, une box (boîte)

est le terme technique en LATEX pour un contenant invisible qui peut contenir

soit un élément visible, soit une autre boîte, soit rien du tout. Ensuite, chaque

boîte est connecté grâce à de la glue (colle), qui détermine la séparation entre

les boîtes.

Dans un document traditionnel, les “lettres-boîtes” sont donc collées à

d’autres pour former des mots, eux-mêmes collés élastiquement à d’autres

mots pour former des phrases. Ces phrases sont découpées en lignes et placées dans un paragraphe (boîte encore une fois), écarté ou collé à d’autres

paragraphes de manière élastique là encore, cette fois pour former des pages

suffisamment aérées et remplies.

C’est donc ainsi que LATEX construit un document et les pages qui le compose, en collant les boîtes ensembles et grâce aux règles de base (natives) et

à celles définies par l’utilisateur.

Concrètement, une boîte ressemble à :

p

hauteur

\height

profondeur

\depth

largeur

\width

ligne de base Indi

Figure 7.1 – Schéma d’une boîte et de ses composants

Les différentes dimensions de la boîte sont contenues au sein de 4 variables : \width pour la largeur de la boîte ; \height pour sa hauteur ; \depth

pour sa profondeur ; et \totalheight pour sa hauteur totale soit \height

+ \depth.

87

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

Ces variables ne peuvent être utilisées que pour des boîtes. Passons à un

peu de pratique pour voir comment appeler les boîtes en question et se servir

de ces dimensions qui leur sont propres.

La pratique

Tout d’abord, les framebox constituent la base en LATEX. La commande

est assez simple :

\framebox[largeur][pos]{texte}

avec les choix suivants pour pos 3

:

❖ l pour aligner texte sur la gauche de la boîte ;

❖ r pour aligner texte sur la droite de la boîte ;

❖ s pour aligner texte sur toute la longueur de la boîte.

Si texte fait référence à l’objet à encadrer (texte, image, formule. . .),

largeur fait référence à la largeur de la boîte. Tu peux renseigner une unité

de distance (13pt, 215mm, etc.).

Naturellement, si tu renseignes 1cm alors que le texte en fait 2, le résultat

risque de ne pas être satisfaisant. Les dimensions propres aux boîtes peuvent

alors être utilisées, comme \width :

Utilisation des framebox

\framebox[1cm]{Texte} \\

\framebox[13pt][r]{Test} \\

\framebox[45mm][s]{Plus grande bo

îte} \\

\framebox[\width]{Pile poil !} \\

\framebox[2\width]{Espacement} \\

\framebox[\linewidth]{Largeur de

la page}

Texte

Test

Plus grande boîte

Pile poil !

Espacement

Largeur de la page

Comme démontré avec le dernier cas de figure, la longueur \linewidth,

présente nativement sous LATEX, peut être utilisée pour créer une boîte de

largeur égale à celle de la page. 4

3. Par défaut, si rien n’est indiqué pour pos, le texte est centré à l’intérieur de la boîte.

4. La notion de « longueur » sous LATEX est abordée plus amplement en page 96.

88

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

Il existe un raccourci pour appeler plus simplement une framebox avec

la commande \fbox. Aucune option n’a besoin d’être indiqué, juste la partie

texte. La taille de la boîte s’adapte alors au contenu :

Le petit raccourci sympathique

\fbox{Un peu de texte} \\

\fbox{Un peu plus de texte} \\

\fbox{Des maths : $i = \sqrt

{169}$}

Un peu de texte

Un peu plus de texte

Des maths : i =

√

169

De l’utilisation des fbox

Les fbox sont très pratiques, pour comprendre comment un

paragraphe ou une image est agencé, ainsi que la taille qu’il occupe.

Lors de montages ou de nouvelles créations, il peut se révéler très

utile d’encadrer les différents éléments grâce à des fbox pour comprendre comment LATEX les agence et pouvoir apporter les correctifs

nécessaires afin d’avoir le résultat souhaité !

Tu trouveras aussi dans la littérature les makebox, dont l’appel est rigoureusement identique à une framebox. Il s’agit tout simplement d’une

framebox sans cadre, ce qui ne présente que peu d’intérêt selon moi.

Techniquement, la framebox est construite à partir d’une makebox mais

j’ai trouvé plus judicieux de présenter directement la première. Bien, terminons avec une autre boîte bien utile.

La plus utile

La boîte qui se révèle bien utile pour faire quelques montages reste la

parbox et l’environnement qui lui est associé : la minipage. Sa syntaxe est

la suivante :

\parbox[ext][hauteur][int]{largeur}{texte}

avec hauteur et largeur respectivement la hauteur et la largeur de la boîte

(distance manuelle comme 13mm ou une longueur propre aux boîtes comme

\width ou une longueur LATEX comme \linewidth).

89

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

Fonctionnement de base d’une parbox

\parbox{13mm}{Texte} \\

\fbox{ % Intérêt de la fbox !

\parbox{\linewidth-2cm}{Partie A

\\ Partie B}

}

Texte

Partie A

Partie B

Ensuite, ext correspond à l’alignement externe de la parbox par rapport

à la ligne de base 5

, avec les choix suivants :

❖ m (par défaut) ou si aucune option n’est donnée pour centrer la boîte

sur la ligne de base ;

❖ b pour aligner le bas (bottom) de la boîte sur la ligne de base ;

❖ t pour aligner le haut (top) de la boîte sur la ligne de base.

Alignement externe

A : \parbox[b]{2cm}{Par. 1 \\ Par. 2} \hfill

B : \parbox{2cm}{Par. 3 \\ Par. 4} \hfill

C : \parbox[t]{2cm}{Par. 5 \\ Par. 6}

A :

Par. 1

Par. 2 B : Par. 3

Par. 4

C : Par. 5

Par. 6

Enfin, int désigne l’alignement interne de la boîte, pour pouvoir positionner verticalement le texte dans la boîte, sous réserve qu’une hauteur

ait été indiquée. Elle peut prendre quatre valeurs :

❖ b pour repousser le texte vers le bas de la boîte ;

❖ t pour situer le texte en haut de la boîte ;

❖ c (par défaut) ou si aucune option n’est donnée pour centrer verticalement le texte ;

5. Ligne sur laquelle reposent les lettres.

90

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

❖ s pour étirer verticalement le texte.

Toutefois, comme tu as pu le constater avec mes exemples, il n’est pas

évident de gérer plusieurs paragraphes dans une parbox. Il est même impossible d’y introduire d’autres environnements !

Il faut donc inclure le tout dans l’environnement équivalent : minipage.

L’appel à cet environnement se fait de la manière suivante, avec des paramètres identiques à ceux d’une parbox, mais dans un ordre différent :

\begin{minipage}[ext][hauteur][int]{largeur}

texte

\end{minipage}

Il convient donc de définir correctement une minipage si tu veux éviter

les erreurs. Dès l’instant où tu renseignes une des options non obligatoires

(ext, hauteur ou int), il faut toutes les indiquer ou le rendu ne sera pas

conforme :

Appel de minipage

\begin{minipage}{0.8\linewidth}

Texte avec un \\

retour à la ligne !

\end{minipage} \\ \\

\fbox{

\begin{minipage}[m][1cm][b]{2cm}

Lorem

\end{minipage}

} \& ipsum

Texte avec un

retour à la ligne !

Lorem

& ipsum

!

De l’utilisation des minipage

Il ne faut pas utiliser une minipage pour simplement avoir

un texte sur 2 colonnes !

D’abord, le résultat ne correspondra pas à tes attentes, ne

sera pas esthétique et sera difficile à gérer. Ensuite, LATEX met à

disposition l’option twocolumn lors de la définition de la classe.

Une minipage sert donc exclusivement pour des montages, par

exemple une image à côté d’une autre image ou d’un texte, comme

91

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

!

nous aurons l’occasion de le voir par la suite.

Enfin, il faut savoir qu’il n’y a pas d’alinéa dans une minipage :

dans sa définition, par défaut, l’indentation est nulle.

Le petit bonus

Si tu veux continuer à arpenter le chemin des boîtes et avoir encore plus

de personnalisation, je te recommande le package fancybox. Il permet, entre

autres, d’ajouter du surlignage et de l’ombrage aux boîtes.

Mais il existe un autre package bien plus puissant. . .

7.5 Le Saint-Graal des boîtes

Découvert lors de la rédaction de la première version de ce guide (été

2016), le package tcolorbox est extrêmement complet et permet une personnalisation totale des boîtes. Tous les encadrés que tu as pu rencontrer

jusqu’à présent dans ce guide sont générés grâce à ce package !

Si tu fouines un peu sur Internet, tu devrais trouver la documentation

officielle. . . allez, je suis gentil, je te donne le lien : http://fr.lmgtfy.com/

?q=tcolorbox+help.

C’est actuellement 500 pages complexes mais qui assez illustrées, pour te

permettre donc de réaliser des boîtes aussi jolies que celles présentes dans ce

guide et bien plus. Beaucoup plus !

Pour te donner un premier aperçu, voici un exemple extrêmement simple :

92

Chapitre 7. Les tableaux et boîtes sous LATEX

;A<

Première utilisation de tcolorbox

% Ajout dans le préambule

%\usepackage{tcolorbox}

\begin{tcolorbox}[colframe =

orange, colback = orange!50,

boxrule = 2pt, arc = 6pt,

title = {Un titre}, coltitle

= black]

J'adore ce package ! \\

De toute mon âme !

\end{tcolorbox}

Un titre

J’adore ce package !

De toute mon

âme !

Et il ne s’agit que la partie émergée de l’iceberg ! La documentation officielle décrit toutes les options disponibles, les différentes boîtesmise à disposition, la création d’environnement pour appeler ses propres boîtes. . .

Mais je crois m’être légèrement emporté. Je reviens sur ce package plus

en détail dans la partie suivante. Le but de cette partie reste de te présenter

les bases sous LATEX.

Passons désormais à un point plus sympathique mais que j’avais envie de

garder pour la fin. Oh, mais je suis persuadé que tu l’attendais depuis un

petit moment : comment insérer une image !

93

Chapitre 8

Insérer des images

U

ne bonne image suffit des fois à remplacer 13 lignes de texte. Découvrons

quelques spécificités à leur sujet et comment en insérer sous LATEX.

Par la suite, pour alléger les exemples, le préambule ne sera plus renseigné

dans les codes LATEX mis à disposition. Ces derniers seront basés sur l’architecture du code minimal fourni ci-après. L’ajout de nouveaux packages sera

signalé au début du code par un commentaire.

Le code minimal

\documentclass[a4paper, 12pt]{report}

% PDFLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\begin{document}

% Ecrire le code ici !

\end{document}

94

Chapitre 8. Insérer des images

;A<

8.1 Les formats d’images

Il existe deux sortes d’images : les images matricielles et les images vectorielles. Les premières sont les plus courantes et portent généralement les

extensions .jpg (Joint Photographic Group) ou .png (Portable Network Graphics). Les secondes sont les plus propres et utilisent des extensions comme

.svg (Scalable Vector Graphics) ou .eps (Encapsulated PostScript). 1

La base d’une image matricielle est le pixel, d’une couleur donnée et

figée. Si tu zoomes sur l’image à la page précédente, tu devrais tomber sur

ces fameux pixels. A priori, rien de méchant : dès lors que ton image contient

“suffisamment” de pixels par rapport à la taille affichée, elle ne devrait pas

apparaître trop floutée.

Une image vectorielle est définie par l’intermédiaire d’outils géométriques

(arcs de cercle, traits, courbes de Bézier, . . .). Je ne vais pas faire un cours

dessus : d’abord parce que je n’en sais pas plus et ensuite parce que ce n’est

pas le but de guide.

Ce qu’il faut retenir c’est que, peu importe à quel point tu zoomes, tu ne

tomberas jamais sur un pixel et l’image vectorielle reste lisse et belle 2

. Et

inversement, si l’image est grande de base, aucun pixel ne sera donc visible.

Cachez cette image matricielle (à gauche) que je ne saurais voir !

Un exemple plus courant d’images vectorielles

Dans un fichier PDF, tu trouves en réalité des images vectorielles

partout. En effet, le texte affiché utilise une police spécifique, définie

vectoriellement.

Et heureusement ! Quel enfer serait sinon la lecture si tout le texte

était flou voire illisible faute d’avoir suffisamment de pixels.

1. La création, édition et visualisation des fichiers .svg sont possibles grâce à des

logiciels spécialisés, comme Inkscape. Le format .eps, moins connu et un peu délaissé de

nos jours, peut être visualisé simplement grâce à des logiciels comme EPS Viewer.

2. J’ai l’impression de faire de la pub’ pour l’Oréal. . .

95

Chapitre 8. Insérer des images

;A<

Le format .eps fait un peu vieux jeu et reste surtout utilisé dans le

domaine scientifique. Cependant, même s’il difficile à modifier avec des outils

standards (comme Paint), il est plus facile à implanter sous LATEX que le

format .svg, pour un résultat identique.

!

Conversion au format .eps

Une image au format .eps n’est pas automatiquement vectorielle.

Supposons que tu ouvres sous GIMP une image matricielle et que tu

l’enregistres au format .eps. Le rendu final reste une image matricielle.

Le format .eps ne garantit pas automatiquement une image vectorielle derrière. Il n’y a pas non plus de transformation miraculeuse

en arrière-plan. C’est bel et bien un format qui peut gérer ce type

d’image mais il ne faut pas s’attendre à ce qu’il fasse de lui-même

une belle conversion.

Pour conserver une véritable image vectorielle au format .eps, il

faut vectoriser l’image matricielle (passage du matriciel au vectoriel),

sous Inkscape par exemple a

, sauvegarder le résultat au format .svg

(sécurité) puis enregistrer cette image vectorielle au format .eps.

a. Cette méthode fonctionne parfaitement pour des formes simples, avec peu de

variations de couleur. Le résultat est à travailler pour des images plus complexes,

voire à créer directement au format vectoriel.

8.2 Les longueurs

Après cette première introduction, nous allons continuer par un petit passage barbare, mais qui va se révéler utile pour la suite. J’en ai déjà brièvement

parlé plus tôt. . . mais c’est l’occasion parfaite pour proprement présenter la

notion de « longueur » sous LATEX.

Sous LATEX, il est possible de travailler avec toutes sortes d’unités : mm,

cm pour citer les plus courantes ; pt, in pour citer quelques cas moins usités ;

ex pour citer l’unité de distance la plus amusante que j’ai découverte à ce

jour en informatique 3

.

Dès lors qu’une commande requiert une longueur en paramètre d’entrée,

nous l’indiquons très clairement. Par exemple, \vspace{13mm}. Cependant,

3. Hauteur d’un « x » : cette unité de longueur dépend donc de la police utilisée.

96

Chapitre 8. Insérer des images

;A<

LATEX permet d’aller plus loin, beaucoup plus loin en mettant des longueurs

prédéfinies sous forme de commande.

Quelques longueurs sous LATEX

LATEX utilise des longueurs nativement, dans chaque nouveau

document. Par exemple, à chaque nouveau paragraphe, LATEX met un

alinéa. La taille de cet alinéa est une longueur définie par défaut et

LATEX utilise sa valeur.

Il en va par exemple de même pour les marges ou les sauts de

ligne. Naturellement, toutes ces longueurs peuvent être modifiables,

même si ce n’est pas vraiment recommandé. C’est aussi ce qui garanti

l’homogénéité (ou la cohérence, si tu préfères) d’un document réalisé

avec LATEX.

Du coup, sans entrer plus dans les détails, voici deux longueurs fondamentales qui sont plutôt utiles :

→ \linewidth : longueur qui correspond à la largeur “locale” du texte

(vis-à-vis de la page, dans un tableau, dans une boîte, etc.) ;

→ \baselineskip : longueur qui correspond à un saut de ligne sous LATEX.

!

\linewidth vs \textwidth

Tu trouveras des fois dans la littérature ou dans des exemples

sur Internet des gens qui emploie la longueur \textwidth. À première vue, lors de son utilisation, elle présente peu de différences avec

\linewidth. Et pourtant, il y a bel et bien une différence ! a

❖ \textwidth représente la largeur d’un bloc de texte (valeur

constante, globale) ;

❖ \linewidth représente la largeur locale du texte, que ce dernier

soit présent dans une colonne, un tableau, une liste ou une

minipage.

En règle générale, il vaut mieux utiliser \linewidth pour spécifier

la taille relative d’une image ou d’une boîte. Cette longueur s’adapte

97

Chapitre 8. Insérer des images

;A<

!

mieux à la situation et aux potentiels montages que tu peux réaliser

(avec des minipage, par exemple).

a. Les points à venir ont été extraits et traduits de la page suivante :

https://tex.stackexchange.com/questions/16942/difference-betweentextwidth-linewidth-and-hsize.

Enfin, il peut être intéressant de savoir qu’un coefficient est toléré devant les longueurs. Par exemple, \vspace{2\baselineskip} correspond à

un double saut de ligne. 0.5\linewidth correspond à une longueur égale à

la moitié de la page (marges exclues).

Voilà, je ne vais pas aller plus loin. Si tu veux en savoir plus sur les

longueurs (création de longueurs, longueurs définies par défaut, etc.), je te

recommande d’aller lire la page suivante : http://en.wikibooks.org/wiki/

LaTeX/Lengths.

Bien, allons maintenant insérer des images. Retiens surtout la longueur

suivante : \linewidth. C’est celle qui va beaucoup nous servir ici.

8.3 Insérer une image

La commande de base

Je pense que tu devais attendre ce point depuis pas mal de temps. Ne

traînons pas plus dans ce cas : place aux insertions d’images !

Travailler avec des images sous LATEX est possible. Il faut au préalable

charger le package graphicx 4

. Pour insérer une image, c’est très simple. Il

faut utiliser la commande suivante, à l’endroit où tu souhaites afficher ton

image :

\includegraphics[options]{nom\_img.format}

Mais je crois qu’un exemple sera plus parlant. Pour ce faire, prends une

image plutôt grande de préférence, soit au format .jpg ou .png 5

, puis

renomme-là fond. De cette manière, tu auras moins de souci avec le code

qui suit. Place cette image dans le même dossier que le fichier .tex avec

lequel tu travailles.

4. Ne pas confondre avec le package de base graphics dont graphicx (avec un « x »

donc) est une version améliorée !

5. Si tu ne connais pas le format de ton image, clic droit puis Propriétés.

98

Chapitre 8. Insérer des images

;A<

Si jamais tu te trompes de format d’images ou que l’image n’est pas

dans le dossier, LATEX va te renvoyer un message d’erreur, du genre « File

nom\_img.format not found ».

Première insertion d’images

% Ajout dans le préambule !!!

%\usepackage{graphicx}

\includegraphics{fond.jpg}

Bon, si l’utilisation de la commande de base est aussi simple, tu conviendras que ce n’est pas très pratique avec une image très grande et qui déborde

pas mal du document ! Voyons donc maintenant comment judicieusement

utiliser les longueurs pour avoir un affichage convenable.

Utilisation des longueurs

Pour ajuster la taille d’une image, 2 options utiles sont disponibles avec

la commande \includegraphics :

1) width = <distance> : forcer la largeur de l’image à <distance>. Cette

option se révèle salvatrice combinée avec la longueur linewidth.

2) height = <distance> : forcer la hauteur de l’image à <distance>.

Essentiellement utile pour des images dont le format « portrait » est

très prononcé, ou si tu veux remplir intégralement la page.

Et c’est tout ce qu’il faut savoir ! Il existe bien une autre option comme

99

Chapitre 8. Insérer des images

;A<

scale mais sans intérêt car la valeur à utiliser dépend de la taille de l’image.

Avec l’option width, peu importe la taille de ton image, elle sera toujours

bien insérée dans ton document. Bien entendu, si ton image reste petite et

matricielle, elle risque d’être floue à l’affichage. Autrement, tu n’as plus à te

soucier de retraiter tes images pour les avoir à une taille appropriée.

Une image bien taillée

% Ajout dans le préambule

%\usepackage{graphicx}

\includegraphics[width =

\linewidth]{fond.jpg}

\begin{center}

\includegraphics[height = 0.25

\linewidth]{fond.jpg}

\end{center}

Comme afficher ci-dessus, tu peux centrer ton image avec un environnement center. La commande \centering fonctionne aussi et va se révéler

utile par la suite.

C’est déjà mieux, non ? Pardon ? Tu voudrais aussi une magnifique légende pour accompagner ton image ? Ta demande est légitime !

Légende et environnement flottant

Tout comme pour les tableaux, l’insertion d’une légende à une image

demande de placer celle-ci dans un environnement flottant. Il s’agit d’une

obligation sous LATEX pour garantir la qualité du document. Tu conviendras

que le rendu ne serait pas très esthétique si l’image était en bas de page et

la légende à la page suivante faute de place.

Visuellement, nous pouvons considérer l’environnement flottant comme

une boîte qui va englober ton image et ta légende et dont la position est

100

Chapitre 8. Insérer des images

;A<

variable, selon la place restante sur ta page :

Environnement flottant

Image

Légende

Figure 8.1 – Schématisation d’un environnement flottant

L’environnement flottant pour les images s’appelle figure. Et comme

LATEX fait bien les choses, il met à ta disposition différentes options pour

positionner correctement cet environnement :

→ t pour top : l’image se retrouve en haut de page ;

→ b pour bottom : l’image se retrouve en bas de page ;

→ p pour page : l’image se retrouve sur une page particulière réservée aux

éléments flottants ;

→ h pour here (le plus pratique) : l’image se retrouve là où elle est

positionnée dans le code.

Cependant, il arrive à LATEX d’être un peu capricieux et l’option “!” devant la lettre lui indique que l’utilisateur a raison. Si tu utilises donc l’option

!h, LATEX fait tout son possible pour placer l’environnement flottant là où il

est placé dans le code.

Ainsi, si le code de l’image est écrit entre une zone de texte A et une autre

zone de texte B, elle le sera aussi sur le document final. . . à condition qu’il

y ait suffisamment de place, naturellement. Dans le cas contraire, l’image

se retrouve à la page suivante et le texte est remonté en conséquence pour

combler les blancs.

101

Chapitre 8. Insérer des images

;A<

Enfin, pour placer une légende, la commande \caption{légende} est

toujours d’actualité pour les images et se place donc à l’intérieur de l’environnement flottant figure.

Environnement flottant (figure) & légende (\caption)

% Ajout dans le préambule

%\usepackage{graphicx}

% La base

\begin{figure}

\includegraphics[width = 0.5\linewidth]{fond.jpg}

\caption{Une première légende}

\end{figure}

% !h + centrage

\begin{figure}[!h]

\centering

\includegraphics[width = 0.5\linewidth]{fond.jpg}

\caption{Une autre légende}

\end{figure}

Si je rencontre quelques soucis pour afficher le résultat du code dans cette

exemple, tu dois constater que la numérotation de la légende est automatisée par LATEX. Parfait, une tâche à laquelle nous n’aurons pas à nous soucier !

Toutefois, certaines complications peuvent parfois se produire avec cette

option !h. C’est pourquoi la solution ultime – que j’utilise constamment –

consiste en l’utilisation du package float et de renseigner un H à la place de

!h. L’image est vraiment contrainte d’être à cet endroit.

Et s’il n’y a pas la place, LATEX laisse un blanc, ce qui laisse parfois un

gros trou dans ton document. . . Difficile d’arriver à tout concilier !

102

Chapitre 8. Insérer des images

;A<

La meilleure solution (selon moi)

% Ajout dans le préambule

%\usepackage{graphicx, float}

% Package float

\begin{figure}[H]

\centering

\includegraphics[width = 0.5

\linewidth]{fond.jpg}

\caption{Avec une légende !}

\end{figure}

Figure 8.2 – Avec une

légende !

!

Disposition des images & marges

Quand une image ne rentre pas en bas d’une page, il peut être

tentant dans un premier temps d’augmenter les marges du document

pour laisser plus de place à l’image. C’est exactement ce qu’il ne

faut pas faire !

Tout d’abord, augmenter les marges permet en effet de gagner

quelques lignes de texte et donc de gagner la place attendue pour

placer l’image. Mais la longueur \linewidth est aussi augmentée en

conséquence donc ton image est plus grande ! Ce n’est donc pas une

solution.

De plus, changer les marges bouscule toute la structure et

l’agencement de ton document, les blancs laissés par les images qui

ne rentrent pas en bas de page (si choix de l’option H avec le package

float). Tu risques donc de perdre un temps considérable à tout

réajuster à chaque fois.

Je recommande donc de procéder de la manière suivante :

1) Régler les marges à la création du document (marges natives

inchangées, choix personnel ou consigne de mise en page du

rapport).

2) Rédiger ton document, inclure les images.

3) Revenir sur ton document, réagencer les images, les réduire,

103

Chapitre 8. Insérer des images

;A<

!

faire des montages, etc. pour limiter les blancs et avoir le

meilleur rendu (subjectif).

Tu peux aussi t’occuper de cette dernière étape chapitre par

chapitre par exemple (changement de page par défaut entre

deux chapitres).

Bien abordons désormais un dernier point capital : la gestion des images.

Bien ranger ses images

Si jamais tu as beaucoup d’images dans ton rapport, tu peux vite noyer

le dossier de travail où se trouve ton fichier .tex.

Dans ce cas, tu peux placer tes images dans un dossier, situé au même

endroit que ton fichier .tex, puis utiliser la commande suivante dans le

préambule :

\graphicspath{{./nom\_du\_dossier/}}

Fais attention à bien placer cette commande après le package graphicx,

car il s’agit d’une commande de ce même package.

Grâce à cette commande au nom assez explicite, tu indiques à LATEX le

répertoire/dossier où tu as rangé tes images. 6 Tu n’es pas limité à un seul

chemin, tu peux en indiquer autant que nécessaire si besoin.

Si le dossier est placé à un autre endroit, la commande s’applique toujours

mais, dans ce cas, il faut renseigner le chemin complet pour accéder jusqu’au

dossier.

!

Nom des images et des dossiers

Le nom de tes images ou des dossiers où tu places tes images ne

doit contenir ni accent ni espace. Autrement, tu risques de ne pas

pouvoir compiler ton document et tu ne vas pas comprendre l’erreur.

Le nom texte mathématiques est donc à bannir. Tu peux

par contre appeler ton image texte\_maths, textemathematiques,

texte-maths, etc.

Bien, voyons maintenant comment faire référence à une image.

6. En informatique, la “commande” ./ fait référence au dossier où se trouve le fichier

avec lequel tu travailles. Pour revenir au dossier parent, il faut utiliser ../.

104

Chapitre 8. Insérer des images

;A<

8.4 Les références

Une image, une équation, un tableau, une partie. . . tous ces outils sont

bien pratiques mais que valent-ils si tu ne peux y faire référence ? Par exemple,

comment écrire automatiquement « cf. l’image no x page y » ?

Le but est bel et bien d’avoir une numérotation automatique : c’est bien

plus pratique et moins fatiguant que de devoir corriger tout ton document

à la main (et même impossible et impensable sur un rapport de plusieurs

centaines de pages).

Naturellement, LATEX propose nativement une solution, ou je n’aborderais pas le sujet. Donc pas de nouveaux packages pour cette fois !

Si tu veux créer une référence, il faut procéder en 2 étapes :

1) Création de la référence avec la commande \label{nom-ref}. Cette

commande est à placer après une légende par exemple (\caption pour

rappel).

Tu peux aussi l’utiliser dans un environnement mathématiques (si tu

veux faire référence à une équation ou dans un paragraphe (pour renvoyer à un bout de texte en particulier).

2) Appel de la référence avec la commande \ref{nom-ref} (numéro de la

légende, de l’équation ou section dans lequel se situe le texte).

La commande \pageref{nom-ref} est disponible nativement (appel du

numéro de page où se situe la référence et donc l’objet référence), tandis

que les commandes \nameref et \autoref sont présentes avec le package

hyperref.

Et si tu veux encore d’autres fonctionnalités, il paraît que le package

cleveref est LA solution. Je dois encore le tester donc je ne vais pas m’épancher sur le sujet.

Voici un petit exemple pour mieux comprendre le fonctionnement des

références :

105

Chapitre 8. Insérer des images

;A<

Faire référence à une image

% Ajout dans le préambule

%\usepackage{graphicx, float}

\begin{figure}[H]

\centering

\includegraphics[width = 0.5

\linewidth]{fond.jpg}

\caption{Légende}

\label{exemple-ref-img}

\end{figure}

Mon image est la \figurename{}

\ref{exemple-ref-img}, située

en page \pageref{exemple-ref

-img}. \\

Avec le package \verb?hyperref? :

\nameref{exemple-ref-img} \&

\autoref{exemple-ref-img}.

Figure 8.3 – Légende

Mon image est la Figure

8.3, située en page 106.

Avec le package

hyperref : Légende

& Figure 8.3.

Comme tu peux le constater, les références se mettent à jour automatiquement. Cette fonctionnalité est très puissante et extrêmement pratique :

tu n’as plus à te soucier de devoir tout mettre à jour manuellement à chaque

ajout d’une image. LATEX a tout en mémoire et l’adapte si besoin.

Pour information, le fonctionnement est similaire pour les formules mais

l’appel de la référence se fait avec la commande \eqref{nom-ref}.

!

Une question ?

« Je ne comprends pas. J’ai compilé et j’ai ?? à la place de mes

références. Pourquoi ? »

Ce n’est rien de grave. C’est le même problème que pour le sommaire. LATEX stocke les références dans un fichier à part à la première

compilation et ne s’en sert que lors de la seconde.

Il faut donc juste compiler deux fois pour afficher correctement les références ou les mettre à jour..

106

Chapitre 8. Insérer des images

;A<

Allez, un peu de courage. Tu touches presque à la fin de ce guide. Tu

pourras alors être autonome sous LATEX, et taper de magnifiques rapports.

Et la suite n’est pas compliquée : c’est du code pour faire des montages

d’images et t’éviter de chercher pendant des heures comme j’ai eu à le faire !

8.5 Un peu de montage

Je suis sûr que, si tu n’y penses pas maintenant, tu souhaiteras à l’avenir faire quelques montages avec des images. Par exemple, placer 2-3 images

l’une à côté de l’autre ou une image avec du texte autour.

Pour ce dernier cas (image avec du texte autour), il existe des solutions,

comme le package wrapfigure qui fonctionne plutôt bien mais qui doit être

utilisé avec des pincettes. Il est fortement recommandé d’aller jeter un coup

d’œil à l’aide en ligne.

Concrètement, pour expliquer le fonctionnement de ce package, il permet

de positionner une image sur la droite ou sur la gauche, dans un bloc de taille

fixée par l’utilisateur. Le texte qui suit la commande épouse alors le contour

de l’image avant de reprendre son cours initial.

Je m’arrache toujours les cheveux à chaque fois que je l’utilise car je

trouve que le rendu n’est jamais à la hauteur et beaucoup de problèmes se

posent dès qu’une légende est ajoutée à l’image. Je ne fournirai donc pas un

exemple ici.

Heureusement, il existe d’autres solutions plus simples comme les minipage.

Si jamais tu as besoin de te remémorer le fonctionnement des minipage, je

te renvoie à la page 91. Sinon, pour aligner côte à côte deux images, il y a

déjà une règle absolument primordiale, ou la compilation ne donnera pas le

résultat espéré :

NE PAS laisser une seule ligne blanche !

Ensuite, la petite recette de cuisine avec les minipage fonctionne de la

manière suivante :

1) Création d’une 1ère minipage de largeur X1\linewidth, avec X1 ∈

]0 ; 1[.

2) Insertion classique de l’image avec la commande \includegraphics et

l’option width. Utiliser la longueur \linewidth (ou une valeur réduite)

devient très pratique dans cette situation.

107

Chapitre 8. Insérer des images

;A<

Par exemple, une image de largeur 0.6\linewidth dans une minipage

de largeur 0.45\linewidth (par rapport à la page ici) sera de largeur

totale 0.27\linewidth, par rapport à la page du coup.

3) Séparation (espace blanc) avec la commande \hfill : remplissage de

l’espace horizontal restant après la création de la 2nde minipage.

Cette séparation permet d’avoir l’image à gauche collée sur la marge

de gauche, celle de droite sur la marge de droite, et d’avoir un beau

séparateur (espace blanc) entre les deux.

4) Création de la 2nde minipage de largeur X2\linewidth, avec X2 ∈]0 ; 1[

et. X1 + X2 < 1 (ou le \hfill n’a aucun intérêt).

5) Si insertion de légende(s), encadrement de toutes les étapes précédentes

par un environnement figure et placement des légendes respectives au

sein de chaque minipage.

Un exemple ici et maintenant et tout sera plus clair :

108

Chapitre 8. Insérer des images

;A<

minipage & montage d’images

% Ajout dans le préambule

%\usepackage{graphicx, float}

\begin{figure}[H]

\begin{minipage}[t]{0.45

\linewidth}

\centering

\includegraphics[width = 0.6

\linewidth]{fond.jpg}

\caption{Lég. 1}

\end{minipage}

\hfill

\begin{minipage}[t]{0.45

\linewidth}

\includegraphics[width =

\linewidth]{fond.jpg}

\caption{Lég. 2}

\end{minipage}

\end{figure}

Figure 8.4

– Lég. 1

Figure 8.5

– Lég. 2

Une fois que tu as saisi le principe pour 2 images, rien ne t’empêche d’en

aligner autant que tu le souhaites, à condition d’avoir la place (ou tes images

risquent d’être très petites).

Tu peux aussi moduler à ta guise la largeur des différentes minipage :

rien ne t’oblige à toutes les avoir de la même largeur, par exemple. À toi

d’adapter cet exemple en fonction de ton besoin !

Il est aussi possible de mettre du texte dans une minipage, pour insérer

une courte explication à côté de l’image. Attention toutefois si le texte

est trop grand : ta minipage va prendre trop de hauteur, le rendu ne sera

plus aussi esthétique et la place risque de manquer.

Dans ces cas-là, il faut soit être synthétique, soit utiliser le package

wrapfig, soit revoir le rendu souhaité.

109

Chapitre 8. Insérer des images

;A<

minipage & texte

% Ajout dans le préambule

%\usepackage{graphicx, float}

\begin{figure}[H]

\begin{minipage}{0.55\linewidth}

J'aime le chocolat !

\end{minipage}

\hfill

\begin{minipage}{0.4\linewidth}

\centering

\includegraphics[width = 0.86

\linewidth]{fond.jpg}

\caption{Légende}

\end{minipage}

\end{figure}

J’aime le chocolat !

Figure 8.6

– Légende

Pour terminer, si tu veux t’amuser un peu, voici un premier aperçu d’une

autre option sympathique intégrée avec le package graphicx :

Yolo !

% Ajout dans le préambule

%\usepackage{graphicx, float}

\begin{center}

\includegraphics[width = 0.5

\linewidth, angle = 13]{fond.

jpg}

\end{center}

Il existe pas mal d’autres options, ainsi que la très pratique commande

\resizebox, que nous aurons la chance de recroiser plus tard dans ce guide.

Mais je te laisse aller lire la documentation officielle 7

. Les explications de

base sont sur ce guide et c’est ce qui m’importe.

Et voilà, tu approches de la fin de ce guide. Tu peux clairement t’arrêter

7. Disponible sur https://ctan.org/pkg/graphicx.

110

Chapitre 8. Insérer des images

;A<

après le chapitre 9 Traitement des erreurs et revenir à ce guide beaucoup plus

tard selon tes besoins.

111

Chapitre 9

Traitement des erreurs

Les erreurs peuvent être nombreuses sous LATEX et pas toujours évidentes

à corriger. Tout d’abord, nous appelons « erreur » en LATEX tout bout de

code qui nuit à la compilation du document et l’empêche de se poursuivre.

Une erreur ne permet donc pas au compilateur de produire le fichier PDF

espéré.

Ensuite, il est important de savoir que toutes les erreurs qui vont être

abordées sont retournées par LATEX, suite à la compilation. Ces erreurs sont

affichées par Texmaker, dans une fenêtre spécifique tout en bas (bouton

« Messages/Log » en bas à gauche pour faire apparaître la fenêtre « Informations du compilateur » si inexistante).

Figure 9.1 – Rappel de l’organisation de l’interface de Texmaker

112

Chapitre 9. Traitement des erreurs

;A<

Quand il y a une erreur, cette fenêtre t’indique aussi la ligne de

code qui pose problème à LATEX pour compiler (colonne Line). 90 %

du temps, c’est sur cette ligne ou dans ses environs qu’il faut relire son code

et chercher l’erreur.

Voyons maintenant la liste des erreurs couramment rencontrées quand tu

débutes avec LATEX, et mes conseils personnels pour les éviter, synthétisée

sous forme d’un tableau.

Les erreurs courantes Comment les corriger

Missing $ inserted 2 cas fréquents :

→ oubli de fermer un mode mathématiques ⇒ le fermer avec

le symbole $ manquant ;

→ emploi d’un symbole propre au

mode mathématiques (^ ou \_

par exemple) ⇒ supprimer le ^

inutile ou utiliser la commande

\\_ pour afficher un underscore

en mode texte.

Missing } inserted

ou

I suspect you have

forgotten a `}'

Très probablement, oubli de fermer une commande par une accolade } ⇒ Commencer par chercher les erreurs parmi les lignes de

codes écrites ou modifiées depuis

la dernière compilation.

Au début, compiler régulièrement son code permet de simplifier la correction de cette erreur. 1

! Too many }'s. Plus rare : oubli d’une accolade

ouvrante. Mêmes conseils que cidessus.

(suite sur la page suivante)

1. C’est plus pratique de corriger plein de petites erreurs que de s’arracher les cheveux

sur un très grand nombre.

113

Chapitre 9. Traitement des erreurs

;A<

Les erreurs courantes Comment les corriger

There's no line to end here Saut de ligne incompris par LATEX

(après un environnement center

par exemple).

⇒ Commencer par regarder le résultat sans saut de ligne : certains

environnements laissent un peu

de blanc avant et après (comme

center justement). Autrement,

utiliser la commande \vspace.

undefined control sequence 2 cas possibles :

→ oubli d’un élément à un endroit, comme une virgule lors

d’un espace insécable (\13 vs

\,13) ;

→ appel d’une commande inexistante ou appel d’une nouvelle

commande bien définie mais

faute de frappe lors de son écriture.

⇒ Vérifier le code et le corriger.

(suite sur la page suivante)

114

Chapitre 9. Traitement des erreurs

;A<

Les erreurs courantes Comment les corriger

Package inputenc Error:

Unicode char, suivi éventuellement d’un caractère et de son code

UTF-8

Utilisation d’un caractère du clavier interdit avec ce moteur de

compilation.

L’exemple le plus courant : symbole ° (commande \degres{} sous

PDFLATEX ; appel “normal” au clavier sous XeLATEX). 2

Erreur fréquente si texte copiercollé d’un autre document (Word,

PDF, page Internet)

⇒ Dans un 1er temps, reprendre

tous les accents et les apostrophes. 3

Option clash for package

<nom\_package>

Conflit entre certains packages.

⇒ Charger les packages dans un

ordre bien précis. (exemple : package xcolor avanr wallpaper).

Extra alignment tab has

been changed to \cr

Erreur dans un tableau : oubli

hautement probable d’indiquer un

changement de ligne (\\).

⇒ Ajouter le \\ manquant.

! [...] \begin{document}

ended by \end{<env>}

ou

! [...] \begin{<env>}

[...] ended by \end{document}

Environnement mal ouvert ou

fermé. Très fréquent à cause de

l’auto-complétion.

⇒ Aller à la ligne indiquée par

l’erreur, regarder l’environnement

concerné, corriger selon le besoin.

Fin du tableau

Voilà dans les grandes lignes les principales erreurs que j’ai recensées

2. Les moteurs de compilation sont abordés dans la partie suivante si tu es intéressé.

3. Dans ce cas, la fonction Remplacer de Texmaker peut se révéler très utile.

115

Chapitre 9. Traitement des erreurs

;A<

jusqu’à présent. Avec l’expérience, tu verras que tu en feras de moins en

moins ou que tu les corrigeras très rapidement.

Sache aussi que tu peux te rendre sur http://fr.wikibooks.org/wiki/

LaTeX/%C3%80\_l%27aide\_! si tu veux des informations complémentaires.

Mon conseil le plus important

Dès que tu ouvres un $ ou un \[ ou un { ou un délimiteur, ferme-le

en suivant. Puis, tu reviens en arrière et tu écris ton code. Le nombre

d’erreurs devrait diminuer.

L’auto-complétion de Texmaker est aussi très pratique pour éviter

ce genre de désagréments.

Par ailleurs, je souhaite revenir sur l’erreur Option clash for package.

Si jamais tu veux tester un nouveau package pour ton rapport ou adapter

un code trouvé sur Internet, ne jamais le faire sur ton document final !

C’est le meilleur moyen de perdre du temps (compilation et adaptation du

code). Il vaut mieux procéder par étapes :

1) Copie du code à adapter sur un nouveau fichier .tex de test, avec juste

les packages absolument nécessaires.

2) 1

ère compilation pour s’assurer que le code copié fonctionne. Suppression des éléments inutiles et/ou gênants pour la compilation (commandes définies par l’utilisateur et non fournies par exemple).

3) Adaptation du code jusqu’à obtention du résultat souhaité.

4) Copie du code final dans ton rapport, ajout du/des package(s) requis,

compilation et gestions des dernières potentielles erreurs.

Tu verras que tu perdras moins de temps à compiler, à étudier le résultat

dans l’affichage Texmaker et tu travailles sur un fichier de test, sans pollué

ton rapport.

La règle absolue avec les packages

Par défaut, toujours charger le package hyperref en dernier ! (sauf

indication contraire : cf. la documentation du package menukeys par

exemple).

116

Chapitre 9. Traitement des erreurs

;A<

Et voilà, la première partie de ce guide est (enfin) terminée. Toutes mes

félicitations si tu es arrivé jusqu’ici ! J’espère avoir pu t’être d’une aide quelconque et que mes explications étaient assez claires.

Ce n’est pas absolument pas évident de débuter en LATEX. Et si je commence à avoir pas mal de repères et d’expériences, la route est encore longue

avant de pouvoir maîtriser les innombrables facettes de ce langage.

Tu trouveras dans la partie suivante mes notes personnelles sur du code

LATEX plus poussé, pour arriver à produire des résultats de plus en plus complexes. Je tenais initialement à les regrouper dans ce guide pour mon usage

personnel mais je me suis rendu compte qu’elles peuvent aussi aider mes lecteurs.

Bon courage pour la suite et, surtout,

n’oublie pas :

LATEX, c’est la vie !

117

Troisième partie

Aller plus loin avec LATEX

118

Préambule – Le retour

C

onnaître quelques notions sous LATEX peut suffire pour écrire des petits

rapports, des fiches personnelles. Mais LATEX permet de réaliser tellement de contenus différents (lettre, rapport, mémoire, livre, article, guide,

présentation. . .) qu’il serait dommage de ne pas en profiter.

Cette nouvelle partie de mon guide se concentre désormais sur des solutions plus sophistiquées, ou parfois juste plus anecdotiques. Tu peux très bien

ne jamais les utiliser et continuer à rédiger tes documents avec les éléments

présentés jusqu’à présent.

Mais, si comme moi tu es de nature curieuse, ce qui va suivre peut se

révéler utile voire intéressant et t’aider à créer des documents de plus en plus

personnalisés et adaptés à ton besoin.

Dans cette partie, je continuerai de m’efforcer à expliquer les notions en

jeu ou les astuces utilisées. Toutefois, si un point ne te semble pas clair, je te

recommande vivement à aller fouiner un peu sur Internet pour comprendre

ce que je fais 4

.

Et comme je reste extrêmement attentionné, voici une petite liste d’endroits très pratiques pour aller chercher de l’information sur LATEX :

→ http://www.ctan.org/ : le site qui centralise tous les packages LATEX

et leur documentation officielle ! Une référence absolue donc ;

→ http://www.xm1math.net/texmaker/doc\_fr.html : l’aide officielle de

Texmaker, qui fournit aussi des indications sur LATEX ;

→ http://www.grappa.univ-lille3.fr/FAQ-LaTeX/ : une FAQ simple

mais bien fournie ;

→ http://fr.wikibooks.org/wiki/LaTeX : un Wiki sur LATEX en français. Sa version anglaise – http://en.wikibooks.org/wiki/LaTeX –

4. La solution peut aussi se trouver parmi les (nombreux) documents d’aide que j’ai

récoltés, mis à disposition sur mon site : https://glf.cl215.fr/.

119

;A<

contient des fois des informations plus précises et complètes ;

→ http://tex.stackexchange.com/ : les forums, c’est cool. Un forum

sur LATEX, c’est encore plus cool. . . à condition de bien formuler sa

demande ;

→ https://www.latextemplates.com/ et https://www.overleaf.com/

latex/templates : parfois, la solution se trouve dans les templates. . .

Par la suite, pour alléger les exemples, le préambule ne sera plus renseigné

dans les codes LATEX mis à disposition. Ces derniers seront basés sur l’architecture du code minimal fourni ci-après. L’ajout de nouveaux packages sera

signalé au début du code par un commentaire.

Le code minimal

\documentclass[a4paper, 12pt]{report}

% PDFLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\begin{document}

% Ecrire le code ici !

\end{document}

Désormais j’ai fini de blablater. Ok pour toi ? Es-tu prêt ? Alors plongeons

un peu plus profondément dans l’univers (fabuleux) de LATEX !

Adrien Bouzigues

I13 CL215

120

Chapitre 10

Les moteurs de compilation

sous LATEX

G

râce à LATEX, l’utilisateur peut rédiger ses documents tout en séparant

le fond et la forme. Cette séparation requiert l’emploi d’un « moteur de

compilation » pour transformer le code en un fichier PDF.

10.1 Présentation des différents moteurs

Jusqu’à présent, j’ai toujours recommandé de compiler avec le moteur

PDFLATEX. Toutefois, si tu es un peu curieux, tu as pu te rendre compte qu’il

existe plein d’autres possibilités pour la compilation rapide avec Texmaker.

Certaines d’entre elles contiennent justement de nouveaux moteurs de compilation LATEX.

Pour rappel, la rédaction d’un document sous LATEX passe par 3 grandes

étapes :

Code

PDF

Rédaction

(en LATEX)

Compilation

(moteur LATEX)

Document prêt

(fichier PDF)

Figure 10.1 – Les 3 étapes pour rédiger un document sous LATEX

121

Chapitre 10. Les moteurs de compilation sous LATEX

;A<

Le moteur de compilation LATEX utilisé (PDFLATEX jusqu’à présent) fonctionne un peu comme une boîte noire. Personnellement, je ne sais pas comment elles fonctionne. Je sais qu’un fichier .tex est inséré en entrée, le moteur

de compilation tourne et fournit en sortie le fichier .pdf espéré.

Moteur de

compilation

Fichier

.tex

Fichier

.pdf

Figure 10.2 – Schématisation d’un moteur de compilation

Si nous devons lister les différents moteurs de compilation, nous pouvons

en relever 3 principaux :

→ le moteur PDFLATEX, utilisé jusqu’à présent ;

→ le moteur LATEX

1

, qui fournit un fichier .dvi qu’il faut convertir par le

choix Dvi -> PS puis PS -> PDF ;

→ le moteur XeLaTeX ;

accompagnés d’autres moteurs annexes avec leur utilité et leur fonction :

→ le moteur BibTEX, pour générer des bibliographies ;

→ le moteur MakeIndex, pour générer des index.

Il en existe d’autres comme LuaLATEX, voire des solutions plus exotiques

avec KaTEX pour inclure des formules LATEX dans du HTML par exemple.

!

Une question ?

« Je ne comprends pas à quoi peuvent bien servir ces nouveaux

modes de compilation. Après tout, PDFLATEX fonctionne très bien

jusqu’à présent. . . »

En effet, tu as tout à fait raison. Et la conclusion de ce chapitre ira

dans ce sens. Mais il existe aussi des classes (powerdot par exemple)

ou des packages (pstricks) qui ne fonctionnent pas sous PDFLATEX.

Il faut donc parfois recourir à un autre moteur de compilation.

1. Moteur éponyme au langage utilisé, attention aux possibles confusions à venir.

122

Chapitre 10. Les moteurs de compilation sous LATEX

;A<

!

LATEX

ou

XeLATEX

Fichier

.tex

Fichier

.pdf

Contraintes

Figure 10.3 – Schématisation de l’utilisation d’un autre moteur de

compilation

Bien, voyons un peu ces contraintes (classes, packages) qui nous obligent

à utiliser un autre moteur de compilation que PDFLATEX.

10.2 Utilisation des différents moteurs de compilation

Selon ce que tu désires comme document final, il te faut utiliser des classes

et/ou des packages spécifiques parfois. Et il s’avère que certains requièrent

l’emploi d’un moteur de compilation bien spécifique.

À l’heure actuelle, pour rédiger un document sous LATEX, j’ai recensé les

cas d’utilisation suivants, avec le(s) moteur(s) associé(s) :

Cas d’utilisation Moteur à utiliser

Générer un PDF “simple”

(rapport normal, avec des commandes “basiques” et des images)

PDFLATEX ou XeLATEX ⇒ images

au format .png ou .jpg

LATEX ⇒ images au format .eps

(convertir les images si besoin)

Changer la police d’écriture Si package de police, PDFLATEX ou

LATEX

Sinon, XeLATEX & package

fontspec

(cf. détails p. 135)

(suite sur la page suivante)

123

Chapitre 10. Les moteurs de compilation sous LATEX

;A<

Cas d’utilisation Moteur à utiliser

Inclure et/ou fusionner des fichiers

PDF dans le document (cf. p. 142)

Package pdfpages

PDFLATEX ou XeLATEX

Générer une bibliographie (cf. p.

152)

PDFLATEX ou LATEX

BibTEX pour la biliographie

Générer un index (cf. p. 160) PDFLATEX ou XeLaTeX ou LATEX

MakeIndex pour l’index

Générer un PDF avec des dessins

(schémas, circuits électriques, diagrammes, etc.)

Package pstricks : LATEX (le plus

rapide) ou XeLATEX

Package tikz (cf. p. 215) :

PDFLATEX (le plus rapide et

simple) ou LATEX ou XeLATEX

Réaliser des présentations (slides) Classe beamer (cf. p. 259) :

PDFLATEX ou LATEX ou XeLATEX

Classe powerdot : LATEX ou

XeLATEX

Fin du tableau

!

Une question ?

« Pour le changement de police, tu mentionnes dans ton tableau

des “packages de police”. Qu’est-ce donc exactement ? »

Il est possible d’appeler un autre package que lmodern dans le

préambule. Par exemple, un package disponible est times, qui donne

accès à une police équivalente au Times New Roman.

Cependant, il y a toujours un risque relatif à l’utilisation d’une

autre police : comment se comporte la police de manière générale ?

qu’en est-il des mathématiques ? etc. Tu peux avoir de mauvaises

surprises.

C’est pourquoi je recommande de toujours utiliser la po124

Chapitre 10. Les moteurs de compilation sous LATEX

;A<

!

lice native de LATEX (Computer Modern, avec le package lmodern).

Si tu dois la changer, c’est vraiment parce que tu dois faire un

rapport et qu’une police spécifique est imposée.

Pour tous les détails supplémentaires, je te renvoie à la page 135.

Insertion d’images au format .eps

Le format .eps n’est pas l’apanage du moteur LATEX. Tu peux très

bien en insérer avec PDFLATEX (conversion automatique en PDF avant

insertion) ou XeLATEX.

Il s’agit d’un format un peu obsolète, utile pour insérer facilement

des images vectorielles (format .svg), par exemple.

10.3 Bilan

Toujours en vie ? Je veux bien croire cette partie un peu technique et indigeste. Elle demande un peu de pratique et il faut faire des essais, rencontrer

des erreurs pour comprendre le fonctionnement et l’utilisation des moteurs

de compilation.

En résumé

Actuellement, tout est possible et envisageable avec le moteur

PDFLATEX. Techniquement, les moteurs les plus modernes sont

XeLATEX et LuaLATEX. Ils constituent un bon substitut à PDFLATEX

le cas échéant. Mais le temps de compilation reste important, encore

plus sur les gros documents (développement et améliorations en cours).

C’est bien pourquoi, depuis le début de ce guide, je recommande d’utiliser le moteur PDFLATEX. C’est le moteur le

plus simple et pratique à utiliser. Il convertit directement ton fichier

.tex en un PDF.

Tu peux normalement tout faire avec : rapports, présentations,

schémas, bibliographie, index, glossaire, nomenclature, insérer des

images et des PDF, etc.

125

Chapitre 10. Les moteurs de compilation sous LATEX

;A<

Enfin, sous Texmaker, le raccourci de « compilation rapide » est

plus développé pour le moteur PDFLATEX : automatisation de la bibliographie avec l’ajout de BibTEX.

Ce n’est par exemple pas le cas pour XeLATEX. Il faut donc lancer les différents moteurs à la main à chaque fois, ce qui se révéler

fastidieux et pénible sur le long terme !

!

Ce qu’il faut bannir !

Avec le recul, la compilation avec le moteur LATEX et la série de

conversion inhérente est peu recommandée.

Cette série de conversions se révèle efficace sur de petits documents mais le passage PS2PDF devient plus long sur des documents

plus importants. Le format .eps est aussi obligatoire pour les images.

Il faut donc aussi convertir toutes les images que tu souhaites insérer

(images généralement obtenues sur Internet donc au format .jpg ou

.png).

Ce moteur se révèle donc long, laborieux et pas toujours pratique :

à bannir autant que possible !

Et pour finir, tu trouveras ci-après (page 128) un diagramme réalisé par

mes soins qui rend un peu plus explicite l’usage des différents moteurs de

compilation, agrémenté de quelques précisions.

126

Chapitre 10. Les moteurs de compilation sous LATEX

;A<

Table 10.2 – Détails sur les fichiers annexes

(a) Principaux fichiers annexes

.aux Références LATEX

.log Rapport compilation

.out Signets du PDF

(package hyperref)

.synctex.gz Visionneuse

Texmaker

.toc Table des matières

(\tableofcontents)

.lof Table des figures

(\listoffigures)

.lot Table des tableaux

(\listoftables)

(b) Autres fichiers annexes

.bbl Bibliographie mise en

forme

.blg Rapport compilation

bibliographie

.idx Index brut

.ind Index mis en forme

.ilg Rapport compilation

index

127

Fichier source

main.tex

Dossier « Images »

.eps .jpg

.pdf .png

Fichiers .tex

(préambule, chapitres, schémas, etc.)

Bibliographie

main-biblio.bib DVI

PostScript (PS) Fichiers annexes (générés par LATEX)

PDF généré

main.pdf

Mise en forme de

la bibliographie

main.bbl

Création index

.ind

PDF fini !

(bibliographie

et/ou index)

OPTIONNEL

appel

cf. chapitre 11

« Structurer ses

documents »

appel

PDFLATEX

XeLATEX

LuaLATEX

(.jpg, .pdf, .png

.eps toléré)

LATEX

(.eps)

dvi2ps

ps2pdf

+

index.idx

moteur MakeIndex

Moteur initial ×2

Moteur initial ?

❖ Si génération d’un glossaire (package glossaries), le fonctionnement serait similaire à celui d’un index.

❖ Si génération d’une bibliographie et d’un index, Texmaker se charge de lancer tous les moteurs (PDFLATEX,

BibTEX et MakeIndex) dans le bon ordre (compilation rapide).

128

Chapitre 11

Structurer ses documents

L

a rédaction de rapports ou de guides peut très vite contenir un nombre

important de lignes (préambule et les commandes définies, chapitres,

contenu). Dès lors, la relecture devient indigeste et difficile. Il est grand de

reprendre le contrôle sur l’organisation de ses documents LATEX !

11.1 Un peu de rangement

Pour commencer, nous supposons que ton projet LATEX tient dans un

dossier racine, identifié de manière relative par la formulation ./.

e ./

e Chapitres

main-biblio.bib

chap1.tex

...

commandes.tex

packages.tex

e Images

image1.jpg

...

main.tex

Figure 11.1 – Arborescence du projet

129

Chapitre 11. Structurer ses documents

;A<

!

Chemin absolu et chemin relatif

Par la suite, nous allons devoir indiquer à LATEX où sont rangés

les différents fichiers .tex ou images à appeler dans le document.

Nous pouvons fonctionner avec un chemin absolu, c’est-à-dire

en indiquant très précisément son emplacement sur le disque. Par

exemple : C:/<user>/Projets/LaTeX/.

Cette méthode présente un inconvénient majeur : tu dois modifier

dans ton code LATEX tous les chemins indiqués si jamais tu changes

de dossier ! C’est pourquoi il faut utiliser un chemin relatif. ./

renvoie au dossier courant, en l’occurrence celui de main.tex (fichier

LATEX principal qui va générer le rapport).

Et s’il faut revenir en arrière dans l’arborescence, “remonter en

arrière d’un dossier” en quelque sorte, la notation ../ existe aussi.

Voyons maintenant les différentes options possibles pour appeler tous ces

fichiers dans notre document source main.tex.

11.2 Commandes disponibles

Pour une fois, je vais faire un travail de traduction car d’autres personnes

ont admirablement bien synthétisé les 3 commandes que je veux vous présenter et leurs propriétés : https://tex.stackexchange.com/questions/

246/when-should-i-use-input-vs-include.

1) \input{./<chemin>/<nom-fichier>} : importation brute du fichier

<nom-fichier>.tex (copier-coller du contenu). Cette commande se révèle donc utile pour renseigner le préambule (chargement des packages)

ou importer du code LATEX ponctuellement.

Par exemple, je pense notamment à des schémas, graphes ou encore

des formules mathématiques alambiquées que tu ne veux pas recopier.

Cette solution garantit alors l’homogénéité de ton code dans ton document : il suffit de changer le fichier .tex d’origine et les changements

s’appliqueront sur tout le rapport.

2) \include{./<chemin>/<nom-fichier>} : importation plus sophistiquée du fichier <nom-fichier>.tex, avec un \clearpage avant et

après. Cette commande prend alors tout son sens pour importer des

parties logiques de ton rapport – chapter principalement, pour ne pas

être impacté par le \clearpage.

130

Chapitre 11. Structurer ses documents

;A<

C’est pourquoi cette commande a toute son utilité au sein du

fichier source main.tex. Des propriétés notables sont à retenir :

❖ un fichier <nom-fichier>.aux est généré. Il contient toutes les

références et la pagination associée, ce qui octroie un léger gain

de temps à partir de la 2ème compilation.

Si tu génères ton rapport entier puis que tu veux travailler que

sur un seul fichier, tu conserves ainsi la pagination finale ;

❖ l’ajout natif de la commande \clearpage empêche l’utilisation de

\include dans le préambule ou pour des petits morceaux de code ;

❖ il est impossible d’utiliser \include dans un fichier déjà appelé

par cette commande (probablement pour éviter les conflits et les

gestions de multiples fichiers .aux).

3) \includeonly{./<chemin>/<nom-fichier1>,...} : génération du rapport en appelant seulement les fichiers indiqués et appelés par \include.

Son appel se fait uniquement dans le préambule ! Il ne faut pas

non plus mettre des espaces dans la commande. La virgule joue le rôle

de séparateur des différents fichiers.

Dans ce cas, si tu souhaites travailler sur une partie bien spécifique

de ton rapport, tu gagnes un coup de boost considérable lors de la

compilation ! En effet, LATEX va compiler uniquement le(s) fichier(s)

indiqué(s), et non le rapport dans sa totalité. C’est peut-être anodin

sur de petits documents mais c’est extrêmement puissant si le rapport

dépasse 30 pages.

Bilan concis

\input est une macro “bas niveau” qui importe le contenu d’un

fichier donné, comme s’il avait été copié-collé manuellement. \include

permet de gérer le fichier comme une unité logique à part entière

(chapter). Cette commande permet aussi de compiler des fichiers bien

spécifiques grâce à \includeonly{fichier1,fichier2,...}, ce qui

garantit un considérable gain de temps à la compilation !

11.3 La pratique

La théorie, c’est toujours très sympathique mais, dans ce guide, il y a des

éléments concrets. Voyons donc tout de suite l’allure que prend notre fichier

131

Chapitre 11. Structurer ses documents

;A<

source main.tex 1

:

Le fichier main.tex

\documentclass[a4paper, 12pt]{report}

% ICI, enlever les % devant les commandes

% (pour éviter les conflits avec l'arborescence Texmaker)

% Chargement des packages et de la configuration utilisés

%\input{./Chapitres/packages}

%\input{./Chapitres/commandes}

%\includeonly{./Chapitres/chap1,./Chapitres/chap3}

\begin{document}

\everymath{\displaystyle}

%\part{Partie I}

%\include{./Chapitres/chap1}

%\include{./Chapitres/chap2}

%\part{Partie II}

%\include{./Chapitres/chap3}

%\include{./Chapitres/chap4}

\end{document}

Comme tu peux le constater, tu dois écrire le chemin relatif – ./Chapitres/

– à chaque fois. Mais, quand tu y réfléchis, tu ne changes pas tes dossiers tous

les 4 matins. Et quand tu le fais, la fonction « Remplacer » est la bienvenue !

De plus, c’est raisonnable quand tu penses au temps de compilation gagné

par la suite. Ce n’est pas la même chose d’attendre 1 seconde (compilation

d’un fichier) ou 4 secondes (compilation d’un gros rapport), surtout si tu dois

souvent compiler pour vérifier le code et le rendu.

1. Pour rappel, il s’agit du fichier à partir duquel est lancé la compilation.

132

Chapitre 11. Structurer ses documents

;A<

La petite astuce Texmaker

Tu dois trouver pénible d’écrire une partie de ton rapport puis

de devoir basculer sur le fichier main.tex pour compiler. Et je te

comprends !

Fort heureusement, Texmaker propose une solution toute simple : le

« document maître ». Cette solution permet de définir le fichier source

main.tex comme référence : toutes les compilations se font alors depuis

ce fichier, même si tu travailles sur un autre !

Pour activer cette option, ouvrir main.tex, aller sur la barre de

menus, choisir Options puis Définir le document courant.... Et

c’est tout. Tu peux même constater que ce choix a bien été pris en

compte. Il te suffit de regarder en bas à droite de la fenêtre Texmaker.

Et si jamais tu veux réaliser des tests ou générer un fichier annexe

alors que tu travailles sur ton rapport, tu peux soit fermer Texmaker

(après avoir sauvegardé ton travail) soit refaire la même procédure et

choisir alors l’option Mode normal.

!

Les noms de fichiers

Pour la clarté de ce guide, j’ai choisi d’appeler les fichiers chap1,

chap2, etc. Toutefois, je ne recommande pas d’utiliser une

telle nomenclature !

En effet, tu dois penser à tout changer le jour où tu veux ajouter

un nouveau chapitre 1 ou 2 (noms de fichiers et commande \include).

Selon moi, il est donc préférable d’appeler explicitement ses fichiers

(chap-texte, page-garde, annexes, etc.).

11.4 D’autres solutions

Les éléments proposés jusqu’à présent constituent une introduction de

ma part. Rien ne t’oblige à suivre à la lettre cette structure. Libre à toi

de créer plus de dossiers pour ranger tes fichiers et images, voire des sousdossiers. L’arborescence proposée peut évoluer très facilement une fois le

concept compris et assimilé.

Tu es aussi libre de créer des fichiers .tex qui te sont propres, comme une

page de garde personnalisée, une introduction, le sommaire, des annexes, ou

133

Chapitre 11. Structurer ses documents

;A<

encore une 4ème de couverture.

Il existe aussi un package qui permet de distinguer dans la commande

le chemin relatif et le nom du fichier à importer : import. Mais le problème

reste le même selon moi, comme il faut indiquer le chemin à chaque fois (à

moins de définir une commande générique, point qui reste à valider).

La page http://blog.dorian-depriester.fr/latex/template-these/

template-complet-pour-manuscrit-de-these peux aussi t’apporter une

aide considérable et te donner de nouvelles pistes à explorer ! Des points intéressants sont abordés, comme générer un mini-sommaire en début de chaque

chapitre, plusieurs bibliographies, utiliser un backref (bibliographie), et bien

d’autres encore.

134

Chapitre 12

Améliorer son texte et sa mise

en forme

S

i tu sais désormais écrire des paragraphes, faire une page de garde simple,

afficher un sommaire ou utiliser des listes sous LATEX, il existe encore

plein d’autres fonctionnalités sympathiques pour compléter ces éléments. Et

je les recense justement dans ce chapitre !

12.1 Changer la police d’écriture

Introduction

Changer de police constitue un vaste débat sous LATEX. Tout d’abord,

pourquoi vouloir changer ? La police proposée ne convient-elle pas ? Ou estce par habitude d’utiliser auparavant une autre police sous Word ? À moins

que ton école impose une police spécifique pour ton rapport ?

Ensuite, quelle police préfères-tu utiliser ? Avec empattement 1

, sans empattement ? Mais surtout, si tu dois écrire des formules mathématiques, la

nouvelle police possède-t-elle les caractères nécessaires sous un format équivalent ? Je pense notamment aux symboles courants X ou Z

, sans parler

des lettres grecques.

Il existe de multiples façons de changer de police sous LATEX : de manière

locale, pour pouvoir juste profiter d’un effet de style ; puis, de manière globale,

donc sur tout le document. Et il faut au préalable connaître la police que tu

aimerais utiliser.

1. Mode de terminaison d’un caractère : http://fr.wikipedia.org/wiki/

Empattement\_(typographie).

135

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Naturellement, si tu tiens à conserver la police native de LATEX, tu sais

déjà ce que tu dois renseigner dans le préambule et tu peux lire la suite pour

te cultiver, sans forcément devoir l’appliquer.

Empattement et machine à écrire

Tout comme il est possible de mettre un bout de texte en gras grâce à

la commande \textbf{<texte>}, tu peux choisir d’enlever l’empattement

(sans serif) d’un bout de texte grâce à la commande \textsf{<texte>}. Tu

peux aussi choisir de le mettre en valeur différemment grâce à la mise en forme

« machine à écrire » (typewriter) avec la commande \texttt{<texte>}.

Si tu veux appliquer un tel changement sur plusieurs paragraphes, tout

comme \textbf{<texte>} a pour équivalent {\bfseries{}<texte>}, nous

avons donc à disposition le \sffamily et le \ttfamily.

Là encore, il faut délimiter les paragraphes concernés avec des accolades {

et } ou utiliser de telles commandes à l’intérieur d’un environnement, ou encore utiliser directement un environnement : \begin{sffamily} fonctionne

parfaitement !

Et si jamais tu veux annuler ces modifications (localement ou sur un paragraphe) et donc retrouver la traditionnelle police avec empattement, les

commandes \textrm{<texte>} et {\rmfamily{}<texte>} sont disponibles.

Enfin, si tu veux faire de telles modifications sur tout le document, tu

peux aussi renseigner une des commandes suivante dans le préambule :

% Avec empattement (défaut)

\renewcommand{\familydefault}{\rmdefault}

% Sans empattement

\renewcommand{\familydefault}{\sfdefault}

% Machine à écrire

\renewcommand{\familydefault}{\ttdefault}

!

Ordre de priorité des commandes

Les commandes globales sont exécutées en premier lors de la

compilation. Elles sont donc supplantées par l’utilisation des envi136

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

!

ronnements, eux-mêmes écrasés par les commandes locales comme

\texsf{<texte>}.

Il faut donc bien réfléchir au résultat souhaité, ainsi qu’à l’ordre

d’utilisation des commandes précédemment décrites pour y parvenir.

Voyons maintenant un exemple pour comprendre le fonctionnement de

ces nouvelles commandes :

L’empattement et la machine à écrire

Texte \textsf{sans empattement}

ou au format \texttt{machine

à écrire}. \\[\baselineskip]

{\sffamily{}C'est amusant à faire

surtout avec plusieurs

paragraphes. \\

Il faut bien tout encadrer avec

des \textrm{accolades !}} \\[

\baselineskip]

\begin{bfseries}

\ttfamily{}Sinon, je peux aussi

la jouer retro sur plusieurs

paragraphes, à l'intérieur d'

un environnement !

\end{bfseries}

Texte sans empattement

ou au format machine à

écrire.

C’est amusant à faire

surtout avec plusieurs

paragraphes.

Il faut bien tout encadrer

avec des accolades !

Sinon, je peux

aussi la jouer

retro sur plusieurs

paragraphes, à

l’intérieur d’un

environnement !

Bien, maintenant que nous avons vu quelques spécificités quant à l’empattement du texte, passons au changement de police.

Les packages de police

Depuis le début de ce guide, je préconise fortement d’utiliser le package

lmodern et de s’y tenir. Il existe toutefois d’autre packages qui peuvent le

substituer et qui permettent donc d’utiliser de nouvelles polices sous LATEX :

bookman, chancery, charter, ebgaramond & ebgaramond-maths (police Garamond), fourier, mathpazo, mathptmx (police Times), newcent, tgbonum,

137

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

etc. (liste non exhaustive).

Les packages times et palatino sont obsolètes et sont donc déconseillés

à l’usage. D’autres peuvent être utilisés, comme helvet ou courier, mais

sans résultat pour ma part (aucun changement de police).

L’utilisation d’un autre package de police que lmodern est donc encore

un sujet délicat, sur lequel il faut être méfiant et faire des essais au préalable.

Tu peux parfois rencontrer des bizarreries, des erreurs insoupçonnées.

Mais si tu veux faire des essais, je recommande particulièrement le site

suivant : http://www.tug.dk/FontCatalogue/. En quelque sorte, il s’agit

d’une bibliothèque qui recense les polices accessibles sous LATEX. Attention à

bien lire l’aide et les indications affichées : certaines polices ne fonctionnent

qu’avec le moteur de compilation XeLATEX !

Enfin, il est possible d’appeler tous ces packages localement 2

grâce à la

commande suivante :

\fontfamily{<code-police>}\selectfont

Quant au choix de <code-police>, tu peux par exemple renseigner ptm,

ce qui est équivalent à utiliser le package mathptmx (police Times).

Une liste détaillée (mais non exhaustive) est disponible en annexes, page

264. Et si tu veux plus d’explications, tu peux te rendre sur le site : https:

//fr.overleaf.com/learn/latex/Font\_typefaces.

!

La petite subtilité

Certaines polices ne font pas la pluie et le beau temps. Par

exemple, tu peux être amené à cumuler les commandes, comme mettre

du texte en gras et en italique. Jusque là, tout va bien, tu peux même

le faire de deux façons :

\textit{\textbf{test}} ou \textbf{\textit{test}}

As-tu déjà utilisé les petites majuscules ? C’est très propre et vraiment agréable à lire. Pour rappel, tu peux utiliser \textsc{Texte}

ou {\scshape{}Paragraphes}. Toutefois, essaye maintenant :

\textbf{\textsc{Texte}} et \textsc{\textbf{Texte}}

2. L’appel peut se faire aussi globalement si la commande est appelée en tout début de

document.

138

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

!

Ben zut alors, aucune petites majuscules avec le package lmodern.

C’est parce que cette police ne gère pas une telle configuration. Changer de police est une solution possible : le package mathptmx(police

Times) est compatible, par exemple.

Fort heureusement, c’est un cas d’utilisation extrêmement rare.

C’était surtout pour te faire toucher du doigt cette petite subtilité.

Dans ce genre de situation, il faut mieux se résigner et faire avec le

package lmodern, pour limiter les problèmes et les complications.

Utiliser une police externe à LATEX

Il n’y a apparemment qu’un seul moyen de pouvoir utiliser une police

d’écriture non disponible nativement (ou via des packages) sous LATEX. Loin

de moi l’idée que la police par défaut me déplaise, bien au contraire.

Toutefois, quand il faut taper un rapport officiel et qu’une police spécifique de Word est imposée – Calibri ou Cambria par exemple, Times reste

disponible avec le package mathptmx –, il n’y a pas d’autres solutions.

Cette solution fonctionne uniquement grâce à une compilation

sous XeLATEX (ou LuaLATEX). Le résultat est à la hauteur de nos attentes :

accents affichés et utilisation de toutes les autres commandes exactement de

la même façon que sous PDFLATEX (mathématiques, images, tableaux. . .).

Elle permet d’utiliser toutes les polices disponibles sur ton ordinateur, y

compris des polices téléchargées et ajoutées manuellement par la suite.

Pour tout faire fonctionner correctement, il faut utiliser le préambule

suivant :

Changer de police

\documentclass[a4paper, 12pt]{report}

% XeLaTeX / LuaLaTeX

\usepackage{fontspec} % Pour le changement de police

\setmainfont{<nom-police>} % Appel de la police (#17)

% Par exemple, <nom-police> = Arial ou Cambria ou Calibri

\usepackage{polyglossia} % Equivalent de babel

\setdefaultlanguage{french} % Paramétrage fran¸cais

% Autres packages

139

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

\begin{document}

Lorem ipsum dolor

\end{document}

!

Nota Bene

La modification de la police d’écriture n’est pas recommandée

dans le cas d’un document qui contient des formules. Les symboles

utilisés par LATEX peuvent ne pas être (ou ne sont généralement pas)

définis dans cette nouvelle police.

Normalement, LATEX générera malgré tout les formules mathématiques avec la police par défaut soit Computer Modern.

Dans le cas d’une modification locale de la police d’écriture, il faut

utiliser la commande \fontspec{<nom-police>}, ce qui donne l’exemple

suivant :

Changement local de police

\documentclass[a4paper, 12pt]{report}

% XeLaTeX / LuaLaTeX

\usepackage{fontspec}

\setmainfont{Arial}

\usepackage{polyglossia}

\setdefaultlanguage{french}

\begin{document}

Lorem ipsum dolor {\fontspec{Cambria}nam dui ligula} nulla

malesuada porttitor !

\end{document}

140

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

!

Quelques petits changements d’habitude. . .

Il faut enfin savoir qu’utiliser ce nouveau préambule n’est pas sans

de légères conséquences sur des commandes usuelles : \degres{} et

\og mot \fg{} sont définies grâce au package babel.

Tu peux décider de l’ajouter avant l’appel de polyglossia, ce

qui permet de conserver \degres{}. Sinon, tu peux directement

écrire le symbole au clavier (sous Windows, + ° ) ] ). Toutefois, les

guillemets français sont mal définis avec ce nouveau package. . .

La seule solution consiste donc à utiliser les commandes suivantes

plus génériques :

Guillemets français avec polyglossia

\guillemotleft{} % Guillemet fran¸cais ouvrant

\guillemotright{} % Guillemet fran¸cais fermant

Bilan

Nous avons donc vu :

→ comment modifier l’empattement du texte ;

→ comment utiliser un package de police sous LATEX ;

→ comment utiliser une police externe à LATEX, en compilant sous XeLATEX

et grâce au package fontspec.

Changer la police peut se révéler amusant pour certaines réalisations personnelles mais il faut passer sous XeLATEX dans le pire des cas, ce qui peut

augmenter le temps de compilation – qui reste raisonnable malgré tout, je te

rassure, tout au plus de l’ordre de quelques dizaines secondes.

Le format par défaut proposé depuis le début de ce guide convient tout à

fait et permet de te démarquer des autres réalisations. Après tout, ne s’agit-il

pas d’une marque de fabrique signée LATEX ?

141

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

12.2 Changer la taille de police

Si tu es amené à changer de police, tu peux aussi faire face à un autre

problème : la taille du texte n’est pas automatiquement la même d’une police à une autre. Certes, il existe des commandes génériques comme \large,

\Large, \LARGE, \huge ou \Huge pour augmenter la taille du texte mais

il peut arriver que ce soit totalement insuffisant. Heureusement, il y a une

commande toute prête dans ce cas :

Changer la taille de police

Lorem ipsum dolor {\fontsize{<taille1>}{<taille2>}\selectfont

{}nam dui ligula} nulla malesuada porttitor !

% <taille1> : taille de la police (13mm, 215pt, etc.)

% <taille2> : espacement entre les lignes

<taille2> remplace alors (localement) la valeur par défaut de la commande \baselineskip. Naturellement, cette commande \fontsize s’utilise

surtout pour faire un gros titre, délimité par un environnement – center par

exemple – pour ne pas affecter le reste du document.

Et si tu ne sais pas quelle valeur choisir pour <taille2>, je ne réfléchis

plus personnellement et prends toujours la moitié de <taille1>, arrondie si

besoin.

12.3 Inclure des fichiers PDF

Tu es pauvre ? Tu ne veux pas payer Adobe et toutes les options inutiles

qu’il propose, dont la fusion de fichiers PDF ? Peu importe : LATEX sait faire

du très bon boulot, gratuitement.

Pour inclure des fichiers PDF, il faut utiliser le package pdfpages et

utiliser la commande :

Inclure un fichier PDF

% Compilation sous PDFLaTeX / XeLaTeX / LuaLaTeX

\includepdf[pages = debut-fin]{<nom-PDF>}

La commande insère le document <nom-PDF> à l’endroit où la commande

est appelée. Le document doit se situer dans le même dossier que le fichier

142

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

.tex. Sinon, il faut préciser le chemin complet du document. Si tu écris

[pages = -], le document entier sera inclus.

Du coup, il ne te reste plus qu’à placer cette commande autant de fois

que le nombre de fichiers à fusionner et le résultat LATEX contiendra donc tes

différents fichiers les uns à la suite des autres. Et si jamais tu veux mélanger les pages, tu peux renseigner l’option pages de la manière suivante par

exemple : pages = {1,3,2,4,6,5}.

Tu peux aussi te servir de cette commande pour insérer une page d’un

rapport PDF dans ton propre rapport ou un tableau Excel exporté en PDF :

c’est des fois plus simples que de tout recopier sous LATEX.

!

Nota Bene

Le paramètre <nom-PDF> ne doit contenir ni espace ni accent.

C’est le même principe que pour les images.

Pour rappel, la compilation doit se faire uniquement avec

PDFLATEX ou XeLATEX (ou LuaLATEX).

Il existe d’autres options, comme faire une rotation à la page (landscape,

angle = <rotation>), ajuster automatiquement le format (fitpaper), etc.

Pour les connaître ou si tu en as besoin, je te laisse consulter la documentation du package : https://www.ctan.org/pkg/pdfpages.

Pour présenter un point plus technique, il est possible de réaliser des

références sur un tel fichier (renvoi de pages) mais il existe une astuce pour

faire en sorte que le renvoi et le numéro de page correspondent.

Pour ce faire, il faut créer un compteur 3

dans le préambule de la manière

suivante : \newcounter{pdfpage}, puis procéder ainsi (utilisation de l’option

pagecommand de \includepdf) :

Référence d’un fichier PDF inséré

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

3. Si tu es curieux, tu peux commencer à approcher la notion de compteur ici : http:

//fr.wikibooks.org/wiki/LaTeX/Programmer\_avec\_LaTeX#Compteurs

143

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{pdfpages} % Le package

\newcounter{pdfpage} % Le compteur

\usepackage{hyperref}

\begin{document}

J'ai beaucoup de texte à écrire \dots{}

% Enlever le % ci-après

%\includepdf[pages = -, pagecommand = {\refstepcounter{pdfpage

}\label{reference}}]{<nom-PDF>}

% pagecommand : faire un peu ce qu'on veut

% refstepcounter : permet le bon renvoi

% label : création de la référence

Je viens d'intégrer un document très important. Pour le

consulter, aller à la page \pageref{reference}.

\end{document}

Si cette solution fonctionne pour un document d’une page, il faut ruser

pour l’appliquer sur un document plus long ou pour différents documents :

Référence de plusieurs fichiers PDF insérés

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{pdfpages} % Le package

\newcounter{pdfpage} % Le compteur

144

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

\usepackage{hyperref}

\begin{document}

%\includepdf[pages = -1, pagecommand = {\refstepcounter{

pdfpage}\label{ref-debut}}]{<nom-PDF>} % Début document

%\includepdf[pages = 2-12]{<nom-PDF>} % Document

%\includepdf[pages = 13-, pagecommand = {\refstepcounter{

pdfpage}\label{ref-fin}}]{<nom-PDF>} % Fin document

Le document inséré est disponible de la page \pageref{refdebut} à \pageref{ref-fin}.

\end{document}

!

Numérotation des PDF insérés

Par défaut, la numérotation des pages se poursuit lors de l’inclusion de fichiers PDF avec la commande \includepdf. Il faut

donc veiller à ce que ces derniers n’en aient pas (pour éviter les

incohérences) ou supprimer l’insertion du numéro de page grâce à

pagecommand.

Cette dernière idée tombe à pic : étudions désormais la gestion des entêtes et pieds de page !

12.4 En-têtes et pieds de page

Par défaut, LATEX propose principalement trois styles pour les en-têtes

et pieds de page. Ces derniers peuvent être appelés n’importe où dans le

document grâce à la commande \pagestyle{<style>} :

→ le style empty : aucune en-tête et aucun pied de page ne seront affichés.

Pour utiliser ce style, il faut donc utiliser la commande \pagestyle{empty} ;

→ le style plain (style par défaut) : seul le numéro de page est affiché, au

145

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

centre du pied de page. Comme c’est le style par défaut, il n’y a aucune

commande à indiquer.

Toutefois, si jamais tu as utilisé un autre style et que tu veux revenir à

celui-ci, son appel se fait donc grâce à la commande \pagestyle{plain} ;

→ le style headings : apparemment, les en-têtes et les pieds de page sont

définis automatiquement selon la classe de document utilisée. Je ne l’ai

encore jamais utilisé jusqu’à présent.

Bon, il faut reconnaître que nous n’allons pas aller bien loin avec ces

maigres possibilités. For heureusement, il existe un package indispensable,

spécialisé dans la personnalisation des en-têtes et pieds de page : le package

fancyhdr.

Ce package permet de définir son propre style pour pouvoir l’appeler

encore plus facilement par la suite à l’intérieur de ton document. Tu peux

le créer grâce à la commande \fancypagestyle et en respectant la syntaxe

générale suivante :

Création d’un style de page

% Dans le préambule de ton fichier .tex

\usepackage{fancyhdr}

\fancyhf{} % Tout effacer

\fancypagestyle{<nom-style>}{

% Définition du style

<en-tête>

<pied>

}

\pagestyle{<nom-style>}

Comme tu peux le constater, tu définis le style et tu lui associes un nom,

ce qui te permet de l’appeler à ta guise par la suite, surtout si tu veux basculer

avec un style natif de LATEX (empty ou plain par exemple).

Quant à la définition du style en elle-même, tu peux définir les en-têtes

et pieds de page respectivement grâce aux commandes :

\fancyhead[<zone>]{<contenu>} % Perso en-tête

\fancyfoot[<zone>]{<contenu>} % Perso pied

146

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Si <contenu> est totalement libre (choix de l’utilisateur), <zone> est défini de la manière suivante :

<zone> Description

L champ gauche pour toutes les pages

LE champ gauche pour les pages paires

LO champ gauche pour les pages impaires

C champ central pour toutes les pages

CE champ central pour les pages paires

CO champ central pour les pages impaires

R champ droit pour toutes les pages

RE champ droit pour les pages paires

RO champ droit pour les pages impaires

Par ailleurs, il existe quelques commandes pratiques pour <contenu> :

❖ \thepage : affiche le numéro de la page courante ;

❖ \thesection : affiche le numéro de la section courante ;

❖ \thechapter avec un document de classe book ou report : affiche le

numéro du chapitre courant ;

❖ \leftmark : avec un document de classe article, affiche le nom de la

section courante ; avec un document de classe book ou report, affiche

le nom du chapitre courant ;

❖ \rightmark : avec un document de classe article, affiche le nom de

la sous-section courante ; avec un document de classe book ou report,

affiche le nom de la section courante.

Enfin, tu peux définir les épaisseurs des traits de séparation grâce aux

commandes :

\renewcommand{\headrulewidth}{épaisseur} % Séparateur en-tête

\renewcommand{\footrulewidth}{épaisseur} % Séparateur pied

147

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Je pense t’avoir bien assommé avec toute cette théorie. Si tu veux connaître

la personnalisation que j’utilise pour ce guide, la voici :

Configuration personnelle de fancyhdr

\usepackage{fancyhdr, fourier-orns} % En-têtes et pieds de

pages

\fancyhf{} % Tout effacer

\fancypagestyle{main}{

\renewcommand{\headrule}{\hrulefill\raisebox{-2.1pt} {

\quad\decofourleft\decotwo\decofourright\quad}\hrulefill}

\fancyhead[L]{\textsc{\nouppercase{\leftmark}}} % Mettre l

'en-tête en small capitals

\renewcommand{\footrulewidth}{0pt}

\fancyfoot[C]{\thepage}

}

Pour info, sache que je n’ai rien inventé. J’ai trouvé le code dans la documentation du package fancyhdr et je l’ai adapté. Comme quoi, tu peux faire

de belles découvertes avec un peu de lecture technique ! Et comme un petit

exemple ne fait jamais de mal, voici un autre exemple, illustré cette fois :

Utilisation concrète de fancyhdr

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{lipsum, fancyhdr, pifont}

\fancyhf{} % Tout effacer

% Autre personnalisation

\fancypagestyle{main}{

\renewcommand{\headrule}{\hspace\*{\fill} \ding{118} \quad

\ding{118} \quad \ding{118} \hspace\*{\fill}}

\fancyfoot[LO]{\thepage\quad\hrulefill}

}

148

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

\begin{document}

\pagestyle{main}

\lipsum[1-3]

\newpage

\pagestyle{plain}

\lipsum[4-6]

\end{document}

❖ ❖ ❖

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit,

vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida

mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique

senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras

viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est,

iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices

bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at,

mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus.

Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget

orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi

auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies

et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet

magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis

natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque

cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat

at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum

massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo.

Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a,

ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem.

Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer

non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus.

Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim.

Vestibulum pellentesque felis eu massa.

1

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae

lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing

elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam

facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet,

enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus

eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis

tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a

faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl.

Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis

lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat

in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis

eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo

lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed

lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas.

Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est.

Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at,

molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed

ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec

luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies

non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus,

egestas vel, odio.

2

!

LATEX et les pages paires & impaires

« Dis donc, je viens d’essayer ton exemple mais l’affichage sur les

pages paires et impaires ne fonctionne pas ! M’aurais-tu menti ? ? ? »

Et oui, cela ne fonctionne pas et je voulais que tu touches du doigt

ce problème, au moins une fois, pour pouvoir en être conscient.

Un document de classe report est, par défaut, considéré pour être

149

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

!

imprimé uniquement en recto, même si tu peux choisir recto-verso au

moment de l’impression (via les options proposées par l’imprimante).

LATEX ne fait donc pas la différence entre les pages paires et impaires.

Pour résoudre ce problème, il faut donc dire à LATEX que ton

document est en recto-verso. C’est possible grâce à une option supplémentaire dans le documentclass : twoside.

Cette option entraîne alors des marges différentes selon les pages

paires et impaires. À la lecture depuis ton écran, le résultat peut

paraître déstabilisant. . . et pourtant, si tu y réfléchis 2 minutes 15,

c’est parfaitement logique si tu veux faire relier ton rapport !

Toutefois, si tu tiens à avoir une feuille centrée, avec des marges

identiques à gauche et à droite, le package geometry constitue alors

la seule solution.

Bon, que me reste-t-il à présenter ? Ah, je sais : une petite astuce, simple

et courte, histoire de ce se reposer un peu.

12.5 Centrer verticalement du texte

À l’heure actuelle, la solution la plus simple et fonctionnelle pour centrer

verticalement du texte – comme un résumé par exemple – repose sur la

commande \vspace\*{\fill} :

Centrage vertical

\vspace\*{\fill}

Paragraphe à centrer

verticalement sur la page

\vspace\*{\fill}

Paragraphe à centrer

verticalement sur la page

Concrètement, \fill est une distance élastique propre à LATEX. Ici, il

s’agit de l’espace blanc restant sur la page, interprété verticalement grâce à

la commande \vspace\* (un \vspace forcé, pour faire simple). Et comme la

commande complète est utilisée 2 fois, LATEX centre verticalement le texte. 4

4. Il s’agit d’une explication “imagée’, pour comprendre le fonctionnement général de

cette formule. Je pense que les notions en jeu sont plus complexes et je ne me suis pas

encore penché sur le sujet.

150

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Tu peux toutefois avoir un résultat plus personnalisable grâce à la commande \stretch{i} à la place de \fill, avec i un nombre strictement

positif.

Centrage simple & personnalisé

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern, lipsum}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\begin{document}

\vspace\*{\fill}

\lipsum[1-2]

\vspace\*{\fill}

\newpage

\vspace\*{\fill}

\lipsum[3-4]

\vspace\*{\stretch{4}}

\end{document}

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit,

vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida

mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique

senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras

viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est,

iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices

bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at,

mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus.

Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget

orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi

auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies

et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet

magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis

natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque

cursus luctus mauris.

1

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat

at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum

massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo.

Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a,

ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem.

Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer

non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus.

Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim.

Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae

lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing

elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam

facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet,

enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus

eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis

tortor vitae risus porta vehicula.

2

Techniquement, la commande \vfill existe et reste bien plus simple à

151

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

écrire. . . sauf que, pour ma part, elle fonctionne une fois sur deux, voire jamais ! Pourquoi ? Je n’ai pas encore trouvé d’explications satisfaisantes à ce

sujet.

S’il faut aussi centrer le texte horizontalement, l’environnement center

convient parfaitement.

La commande \hfill doit plutôt être utilisée pour équilibrer l’espacement entre des blocs, comme des minipage ou l’intérieur d’un tableau par

exemple.

En cas de problème ou si hfill ne fournit pas le résulta escompté, tu peux

aussi forcer le centrage horizontal avec les commandes \hspace\*{\fill} ou

\hspace\*{\stretch{i}}.

12.6 Générer une bibliographie

Comme LATEX peut vraiment tout faire, en ce qui concerne le traitement

de texte, pourquoi se priver et ne pas générer facilement de (magnifiques)

bibliographies ?

Si je commence à avoir un peu d’expérience avec les bibliographies, je

n’en réalise pas tous les jours pour en comprendre toutes les subtilités. Si

mes explications te semblent un peu confuses, n’hésite pas à te référer aux

sites suivants pour saisir les bases et compléter mes explications :

→ http://www.tuteurs.ens.fr/logiciels/latex/bibtex.html ;

→ http://www.xm1math.net/doculatex/bibtex.html.

Création de la base de données bibliographiques

Sous LATEX, la première étape est la création à proprement parler de ta

bibliographie, sous la forme d’une “base de données”. Pour cela, ouvrons notre

éditeur LATEX (Texmaker) puis créons un nouveau document. Là, renseignons

les lignes suivantes qui nous serviront de test par la suite :

Une première bibliographie

@Article{Johnson,

author = {Edgar G. Johnson and Alfred O. Nier},

title = {Angular Aberrations in Sector Shaped Lenses},

152

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

journal = {Physical Review},

year = {1953},

volume = {91},

number = {1},

}

@Phdthesis{Zoran,

author = {Zoran Racic},

title = {\'Etude et essais du spectromètre à plasma},

publisher = {Université Pierre et Marie Curie},

year = {1996}

}

@Misc{opensource,

author = {{Open Source Initiative}},

title = {The Open Source Definition},

howpublished = {\url{http://opensource.org/osd}},

note = {accès le 10/10/2017}

}

Ensuite, il faut enregistrer le fichier. Ce doit sûrement être un peu la

même recette pour tous les éditeurs de texte en général. Sous Texmaker, il

faut procéder ainsi :

1) Barre de menus.

2) Fichier puis Enregistrer sous.

3) Donner un nom au fichier, comme biblio\_type.bib.

L’extension .bib correspond au format employé sous LATEX pour gérer

une base de données bibliographique. C’est très important car, lors de la

compilation, LATEX va chercher un fichier .bib, le lire et créer la bibliographie

en conséquence.

Le conseil personnel

Je recommande de conserver le fichier biblio\_type.bib dans un

dossier à part en tant que template. Ainsi, tu n’auras pas à le créer à

chaque fois mais juste à faire un copier-coller du fichier en question.

153

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Ensuite, je ne vais pas m’étendre sur les diverses possibilités de

renseigner une bibliographie (article, thèse, livre, site Internet. . .).

Internet fournit suffisamment d’exemples et Texmaker propose des

commandes à trous : dans la barre de menus, choisir Bibliographie

puis Bibtex puis sélectionner le type de document à renseigner pour

la bibliographie.

Personnellement, je viens d’en apprendre un peu plus sur les bibliographies grâce à des explications développées dans l’ouvrage LATEX

- How To, disponible au format numérique ici même : http://www.

latex-howto.be/book/download\_fr (chapitre 10 pour les bibliographies).

Si notre base de données bibliographiques est désormais créée, il peut être

intéressant de comprendre son fonctionnement et la syntaxe employée dans

les fichiers .bib :

❖ une entrée dans la base de données bibliographique commence toujours par un arobase @, suivi du nom du document (parmi ceux disponibles), comme Article ou Phdthesis dans le cas présent ;

❖ le premier élément qui arrange juste après – hormis les accolades de

séparation obligatoires – est un mot, qui va servir de clef, de référence

pour appeler cette entrée bibliographique par la suite.

Comme pour les étiquettes, il est donc interdit de mettre des espaces

ou ces accents !

❖ ensuite, il faut renseigner les différents champs de ton entrée bibliographique (auteur, éditeur, titre, année, etc.), avec les données de chaque

champ entre accolades et une séparation des champs par une virgule.

Certains champs sont obligatoires, d’autres optionnels. Si tu ajoutes

une entrée bibliographique par l’intermédiaire de Texmaker 5

, ces derniers sont facilement repérables : ils commencent par OPT ;

❖ enfin, s’il y a plusieurs auteurs, il faut bien les séparer par un and, et

non une virgule ou un & (esperluette).

Et si jamais il y a un prénom ou un nom composé, il faut “dou5. Pour rappel, dans la barre de menus, choisir Bibliographie puis Bibtex puis sélectionner le type de document à renseigner pour la bibliographie.

154

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

bler les accolades” pour bien séparer le prénom du nom ou le formatage final lors de la création te semblera bizarre ! Par exemple :

author = {André {Mouche Baie}}.

Génération de la bibliographie

Bon, nous avons créé la bibliographie. Il faut maintenant indiquer à LATEX

de la générer et de l’introduire dans le document. Pour ce faire, il faut utiliser les commandes suivantes, dans cet ordre et à l’endroit où doit

apparaître la bibliographie :

→ \bibliographystyle{smfplain} : pour générer la bibliographie avec

des normes françaises. Sans le smf, la bibliographie est générée selon

des normes américaines. 6

Par exemple, le and dans le fichier .bib pour séparer les auteurs reste

tel quel avec le format américain plain, alors qu’il est remplacé par un

et avec le format français smfplain (entre autres modifications donc) ;

→ \bibliography{<nom\_fichier>} : pour indiquer le fichier .bib qui

contient notre bibliographie. Ici, <nom\_fichier> = biblio\_type.

Puis vient la compilation. Il faut alors procéder de la manière suivante :

1) Lancer la compilation sous PDFLATEX, pour la première compilation

du document.

2) Lancer BibTEX, pour la génération de la bibliographie dans des fichiers

annexes (cf. les options Texmaker pour connaître le raccourci clavier

associé).

3) Relancer PDFLATEX deux fois, pour l’intégration de la bibliographie et

la bonne implémentation des références et du sommaire.

Le conseil personnel

Changer la « compilation rapide » de Texmaker avec un choix qui

intègre le moteur BibTEX simplifie aussi grandement la vie.

6. Il existe d’autres formats français de bibliographie mais smfplain constitue une

bonne entrée en matière. SMF est l’acronyme de Société Mathématique de France.

155

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Maintenant, si tu as lancé la compilation, tu dois te dire que c’est nul car

rien n’apparaît. . . et c’est normal ! LATEX ne va pas générer une bibliographie

si tu n’y fais pas référence.

Te souviens-tu de la clef/étiquette créée lors de la création de ta base de

données .bib ? C’est ici qu’elle entre en jeu avec la commande \cite{<clef>}.

Il te suffit donc de placer cette commande à l’endroit où tu souhaites faire

une référence à ta bibliographie.

Et voilà, c’est tout. Il ne faut pas en savoir plus pour générer une bibliographie. Tu as donc désormais toutes les cartes en main pour t’y mettre et

expérimenter.

Faire fonctionner hyperref avec la bibliographie

Il peut être frustrant que les liens dans la bibliographie ne soient

pas coupés, ou qu’un clic sur un élément cité ne renvoie pas à sa ligne

dans la bibliographie.

Pour ce faire, il faut charger les packages ci-après, dans l’ordre

suivant (ou il y aura des conflits entre les packages et des erreurs de

compilation) :

hyperref et la bibliographie

\usepackage[nottoc, notlof, notlot]{tocbibind} % Pour

inclure la bibliographie dans le sommaire

\usepackage[hyphens]{url} % Pour couper les urls dans la

bibliographie

\usepackage[breaklinks]{hyperref} % Inclure les autres

options déjà abordées

\usepackage[hyphenbreaks]{breakurl} % Idem package url

Du coup, pour résumer tout ce qui a été vu durant cette partie, ton fichier

.tex doit ressembler à :

Bibliographie : code complet

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

156

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage[nottoc, notlof, notlot]{tocbibind} % Pour inclure

la bibliographie dans le sommaire

\usepackage[hyphens]{url} % % Pour des césures correctes dans

les URLs, y compris dans la bibliographie

\usepackage[pdfauthor = {{Prénom Nom}}, pdftitle = {{Titre

document}}, pdfstartview = Fit, pdfpagelayout = SinglePage

, pdfnewwindow = true, bookmarksnumbered = true,

breaklinks, colorlinks, linkcolor = red, urlcolor = black,

citecolor = cyan, linktoc = all]{hyperref} % Renvois --

Options Adobe/lecteur PDF

\usepackage[hyphenbreaks]{breakurl} % Idem package url

\begin{document}

Mon document se réfère à \cite{Johnson, Zoran} mais des ré

ponses sont aussi disponibles sur Internet \cite{

opensource}.

% Possiblité de cumuler les \cite dans une même commande

\bibliographystyle{smfplain}

% Enlever le % ci-après

%\bibliography{biblio\_type}

\end{document}

Tous ces éléments te semblent compliqués ? C’est bien normal au début.

Là encore, il faut comprendre que tu sépares le fond de la forme : tu crées une

base de données bibliographique, totalement désordonnée, puis LATEX affiche

uniquement les éléments auxquels tu fais référence.

En parallèle, LATEX gère automatiquement la mise en forme de la bibliographie et classe de lui-même les auteurs par ordre alphabétique ! Pas mal,

non ? C’est donc un peu compliqué au début mais c’est extrêmement puissant.

157

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Ajout manuel d’une bibliographie

Toutefois, si jamais tu n’as que 2-3 références à faire, il existe une manière

plus simple de générer une bibliographie. Tu peux la créer toi-même grâce à

l’environnement thebibliography.

Il suffit alors de compiler seulement 2 fois avec PDFLATEX pour bien intégrer les références et c’est tout ! Nul besoin désormais de passer par BibTEX.

Je te laisse reprendre le code ci-après et digérer les commentaires qui

suivent :

Ajout manuel d’une bibliographie

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\begin{document}

La bibliothèque propose trois livres \cite{latexpratique,

texbook, latexcompanion} : les livres \cite{latexpratique,

latexcompanion} traitent de \LaTeX{} ; le livre \cite{

texbook} traite de \TeX{}.

\begin{thebibliography}{KNU90}

\bibitem{latexpratique} Christian \textsc{Rolland}. \emph{

\LaTeX{} par la pratique}. O'Reilly, 1999.

\bibitem[KNU90]{texbook} Donald E. \textsc{Knuth}. \emph{The

\TeX{}book}. Addison-Wesley, 1990.

\bibitem{latexcompanion} Frank \textsc{Mittelbach} et Michel

\textsc{Goosens}. \emph{The \LaTeX{} Companion}. AddisonWesley, 2004.

\end{thebibliography}

\end{document}

158

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Pour comprendre ce qu’il se passe, voici quelques explications :

❖ début de la création manuelle de la bibliographie avec l’appel suivant :

\begin{thebibliography}{<affichage>}.

<affichage> est une notion assez particulière que je vais détailler ciaprès. Elle reste toutefois optionnelle mais il faut laisser les doubles

accolades {} après l’appel de l’environnement thebibliography, sous

peine de rencontrer des erreurs lors de la compilation ;

❖ création d’une entrée dans la bibliographie grâce à la commande suivante : \bibitem[<réf>]{<clef>}, suivie de l’entrée en question (titre,

auteur, etc.).

Ici, aucun formatage n’est appliquée (titre en italique, nom de l’auteur en majuscules ou en petites capitales. . .) donc tu dois le faire et

respecter les règles en vigueur !

<réf> est lui aussi optionnel. Il permet de mettre un nom comme référence à la place de la numérotation, utilisée par défaut.

Et c’est maintenant qu’il y a une subtilité à saisir ! Si tu choisis d’utiliser

une <réf> de ton cru, comme KNU90 dans l’exemple fourni, tu peux constater

que l’alignement dans la bibliographie n’est pas des plus esthétiques.

Intervient alors <affichage> : il faut renseigner la valeur la plus longue

(en terme de nombre de caractères) parmi les <réf> définies. LATEX procède alors au décalage de toutes les entrées dans ta bibliographie : tout est

désormais bien aligné !

Bilan : de la bonne création d’une bibliographie

Pourquoi se priver d’un tableau synthétique pour résumer toutes ces nouvelles notions ? Comparons donc l’usage d’un fichier .bib à l’environnement

thebibliography.

.bib thebibliography

Base de données .bib Dans le fichier .tex

Fichier supplémentaire à gérer Dans le code

Gestion automatique de la mise

en forme (remplissage de champs

dans une “base de données”)

Mise en forme par l’utilisateur

(suite sur la page suivante)

159

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

.bib thebibliography

Classement par ordre alphabétique

des auteurs (par nom)

Entrées bibliographiques affichées

dans l’ordre de leur création

Cohérence de la bibliographie (affichage des références citées)

Affichage de toute la bibliographie

créée

Compilation sous PDFLaTeX + inclure Bibtex

Tous les modes de compilation tolérés, pas besoin de Bibtex

Fin du tableau

Pour conclure, il n’y a pas de bonnes ou de mauvaises méthodes. Tout

dépend uniquement de l’utilisation que tu dois en faire.

Personnellement, pour une courte bibliographie, je la ferai désormais à

la main, quitte à la générer avec Bibtex initialement pour voir la mise en

forme, comme c’est conçu pour.

Sinon, à partir de 4-5 ouvrages à citer, je ne réfléchis plus et j’implémente

tout dans un fichier .bib.

Allez, passons à la cerise sur le gâteau : la génération d’un index. Ce serait

dommage de s’en priver !

12.7 Générer un index

Définition d’un index

«

L’index est un outil du livre qui consiste en une liste organisée d’éléments appelés termes (mots, concepts, objets. . .) jugés pertinents pour

le lecteur, accompagnés de leur adresse – c’est-à-dire la place où ils sont

évoqués dans l’ouvrage. Il permet au lecteur de localiser rapidement un

élément dans l’ouvrage, sans être contraint de le lire intégralement.

»

Wikipédia – Disponible sur :

https://fr.wikipedia.org/wiki/Index\_terminologique

Pour être un poil plus précis, l’organisation d’un index se fait dans l’ordre

alphabétique. Tu conviendras que c’est quand même nettement plus pratique

pour rechercher une information.

160

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Et pour cette fois, je ne vais pas présenter le cas minimal qui fonctionne

sous LATEX car le résultat est relativement moche à mon sens. C’est pourquoi

je vais plutôt directement te décrire la solution finale, entièrement personnalisable. C’est parti !

Préambule et paramétrage

Pour commencer, il faut charger le package imakeidx qui permet de personnaliser l’index, et d’en gérer plusieurs si besoin, comme nous le verrons

par la suite.

Par défaut, le package lance la compilation intermédiaire avec le moteur

MakeIndex, pour créer l’index avant de l’intégrer au document. Il n’y a pas

besoin de compiler une seconde fois le document pour une fois – mais tu es

toujours contraint de le faire si tu as des références ou un sommaire.

Juste après avoir chargé ce package, il faut indiquer à LATEX de préparer l’index, grâce à la commande \makeindex et les options recommandées

suivantes :

\makeindex[title = {<titre\_index>}, options = {-s index-style.

ist}]

index-style.ist est un fichier spécifique aux index sous LATEX et qui

permet de le styliser grâce à une syntaxe spécifique 7

. LATEX va donc utiliser

ce fichier. . . à condition qu’il existe ! Il va donc falloir le créer.

Si l’environnement filecontents est disponible par défaut sous LATEX, il

est contraignant pour 2 raisons : il doit être placé avant le \documentclass

et il permet de créer un fichier mais ne le remplace pas s’il a été modifié

entre temps. Il faut donc constamment supprimer le fichier manuellement si

tu veux faire des tests pour personnaliser ton index à ta guise.

Fort heureusement, il existe le package éponyme qui permet d’éviter tous

ces tracas et solutionne ces problèmes. Je recommande donc de charger les

lignes suivantes dans le préambule pour commencer :

7. L’extension .ist signifie probablement « index style ».

161

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Personnalisation de l’index : fichier de style

\usepackage{filecontents}

\begin{filecontents\*}{index-style.ist}

% Définition "en-tête"

headings\_flag 1

heading\_prefix "\\vspace{26pt}{\\bfseries\\huge{}"

heading\_suffix "}\\vspace{13pt}"

% Cas particuliers

symhead\_positive "Symboles"

symhead\_negative "symboles"

numhead\_positive "Nombres"

numhead\_negative "nombres"

% Style entre une entrée et le numéro de page

delim\_0 "\\hspace{6pt}\\dotfill\\hspace{6pt}"

delim\_1 "\\hspace{5pt}\\dotfill\\hspace{5pt}"

delim\_2 "\\hspace{4pt}\\dotfill\\hspace{4pt}"

\end{filecontents\*}

Libre à toi de te contenter de ma feuille de style personnelle, ou d’en

développer une qui sied plus à ton goût. Tu trouveras toutes les possibilités

de personnalisation sur le site : https://www.overleaf.com/learn/latex/

Indices (aller à la fin de l’article).

La syntaxe est très simple : renseigner l’élément à personnaliser (cf. le

lien précédent avec la liste complète), délimiter le groupe de commande pour

la personnalisation par des apostrophes ("<cmde(s)>") et enfin, le plus important, doubler les backslash pour appeler une commande LATEX.

Bonus personnalisation

J’ai eu à développer une solution pour colorer le numéro de page de

chaque entrée d’un index, pour le faire ressortir et faciliter la lecture.

Après de nouveaux essais, je me suis rendu compte que cette personnalisation ne fonctionne pas sans le package hyperref, et se révèle

même inutile si l’option colorlinks est utilisée – mais reste utile avec

l’option hidelinks, même si je ne recommande pas son utilisation.

De plus, les couleurs pour les références, définies par l’intermédiaire

162

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

du package hyperref, restent prioritaires sur cette personnalisation.

Je reporte donc ici le code initialement utilisé, en aparté. Peut-être

qu’il fera gagner du temps à quelqu’un ou pourra servir dans un autre

contexte :

Personnalisation (inutile) des références de l’index

% Mise en rouge du numéro de page

encap\_prefix "{\\color{red}\\"

encap\_infix "{"

encap\_suffix "}}"

Pour terminer sur la personnalisation de l’index, je recommande de charger le package idxlayout avec les options totoc et unbalanced, afin d’inclure par la suite l’index dans le sommaire et de garantir un affichage sur une

colonne sur la dernière page de l’index.

Tu penses peut-être que je suis tatillon mais tu verras avec le temps qu’un

index sur 2 colonnes est le plus pratique et lisible. Malheureusement, sans ce

package et cette option, la dernière page peut avoir un affichage curieux.

Ce package propose d’autres options, comme le choix du nombre de

colonnes, la distance entre chaque colonne ou encore l’ajout d’un séparateur (trait vertical d’épaisseur paramétrable) entre tes colonnes (columns,

columnsep et rule). Pour plus de détails, se référer à la documentation officielle du package : https://ctan.org/pkg/idxlayout.

Création d’entrées et de l’index

Maintenant que nous nous sommes occupés de la personnalisation de l’index, voyons comment ajouter des entrées à celui-ci.

Pour ajouter une entrée à ton index, il suffit de la placer à l’endroit

souhaité dans le document avec la commande \index{<entrée>}. Toutefois, il faut savoir que le classement alphabétique du moteur MakeIndex ne

fonctionne pas avec les accents. Il faut donc ruser avec un @, de la manière

suivante :

163

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Création d’une entrée accentuée dans l’index

\index{<entrée\_sans\_accent>@<entrée\_avec\_accent>}

Il est possible de créer une sous-liste dans l’index avec un ! :

Création d’une sous-liste dans l’index

\index{<entrée\_principale>!<entrée\_secondaire>}

Il existe aussi d’autres possibilités de formatage d’une entrée :

Autres options disponibles avec \index

% Indexage de pages multiples

\index{Mecanique quantique@Mécanique quantique!Histoire|(}

En 1901, Max Planck

... \newpage

et c'est ainsi que se termine l'histoire !

\index{Mecanique quantique@Mécanique quantique!Histoire|)}

% Formatage du numéro de page

\index{Cochon|textbf}

% Cumul de commands imposibles en l'état

% Personnaliser le fichier .ist en conséquence si besoin

% Créer un "renvoi" à une autre entrée (manuel)

\index{Pierrot|see{Pierre}}

% Créer un "renvoi" à d'autres entrées (manuel)

\index{Pierre|see{Pierre, diminutif}}

% Autre possibilité de "renvoi"

\index{Jen|seealso{Jenny}}

Enfin, il faut dire à LATEX de générer l’index. Pour cela, il faut juste

utiliser la commande \printindex{} à l’endroit voulu, un peu comme pour

le sommaire ou la bibliographie. Et voilà, c’est tout. Ok pour toi ? Voyons le

résultat final sur un exemple concret dans ce cas.

164

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Générer un index

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{xcolor}

\usepackage{imakeidx}

\makeindex[title = {Index test}, options = {-s index-style.ist

}]

\usepackage[totoc, unbalanced]{idxlayout}

% Index par défaut sur 2 colonnes

\usepackage{filecontents} % Pour pouvoir réécrire directement

sur le document + définition dans le préambule possible (

et non avant documentclass)

\begin{filecontents\*}{index-style.ist}

headings\_flag 1

heading\_prefix "\\vspace{26pt}{\\bfseries\\huge{}"

heading\_suffix "}\\vspace{13pt}\\nopagebreak{}"

symhead\_positive "Symboles"

symhead\_negative "symboles"

numhead\_positive "Nombres"

numhead\_negative "nombres"

delim\_0 "\\hspace{6pt}\\dotfill\\hspace{6pt}"

delim\_1 "\\hspace{5pt}\\dotfill\\hspace{5pt}"

delim\_2 "\\hspace{4pt}\\dotfill\\hspace{4pt}"

\end{filecontents\*}

\usepackage[colorlinks]{hyperref}

\begin{document}

\tableofcontents{}

\vspace{2\baselineskip}

165

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Texte\index{Texte} \\

Autre essai \index{Essai a confirmer@Essai à confirmer} et j'

ajouterai aussi \index{Ajout!Ajout1} que les combinaisons

sont possibles \index{Ajout!Ajout teste@Ajout testé} !

% Pour ajouter un petit texte au début de l'index

\setindexprenote{Ceci est mon index ! \\}

\printindex{}

\end{document}

Table des matières

Index test 2

Texte

Autre essai et j’ajouterai aussi que les combinaisons sont possibles !

1

Index test

Ceci est mon index !

A

Ajout

Ajout testé . . . . . . . . . . . . . . . . 1

Ajout1 . . . . . . . . . . . . . . . . . . . . 1

E

Essai à confirmer . . . . . . . . . . . . . 1

T

Texte . . . . . . . . . . . . . . . . . . . . . . . . . 1

2

!

Texte au début de l’index

Le package imakeidx fournit la commande

\indexprologue{<texte>}, qui ne fonctionne plus suite à

l’utilisation du package idxlayout. C’est pourquoi il faut

employer la commande associée à ce dernier package, soit

\setindexprenote{<texte>}.

166

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

!

Le package idxlayout fournit alors l’option notesep pour personnaliser l’espacement entre ce texte et le début de l’index le cas

échéant. C’est en tout cas plus pratique et plus propre que d’indiquer

un saut de ligne \\ manuellement à chaque fois.

Un autre moteur utile pour générer un index

Si MakeIndex est le moteur historique pour générer des index,

d’autres solutions ont vu le jour, notamment pour prendre en compte

les mots accentués lors du tri alphabétique de l’index.

Le moteur xindy (ou texindy pour la version adaptée à LATEX)

en fait partie. Son utilisation devient apparemment indispensable

si l’index est automatisé par l’intermédiaire de commandes, avec la

présence d’entrées accentuées.

Le package imakeidx propose justement l’option xindy, pour générer l’index avec ce moteur texindy. Malheureusement, rien ne fonctionne actuellement de mon côté a

. . . et MakeIndex me convient très

bien pour le moment !

Si jamais tu veux plus d’informations, j’ai découvert ce nouveau

moteur par l’intermédiaire du site suivant : https://geekographie.

maieul.net/169.

a. Moteur texindy « not found » apparemment, alors que MiKTeX montre qu’il

est bien installé selon toute vraisemblance.

Générer plusieurs index

Grâce au package imakeidx, il est possible de générer plusieurs index. Il

suffit juste de leur donner un nom (option name = <nom\_index>) lors de la

mise en place de l’index dans le préambule avec la commande \makeindex,

puis d’indiquer ce même nom lors de l’ajout d’une entrée de la manière suivante : \index[<nom\_index>]{<entrée>}.

Dans le même esprit, l’appel de l’index se fait alors de la manière suivante : \printindex[<nom\_index>{}.

Sans refaire un exemple concret avec un aperçu du résultat, voici un cas

simple d’utilisation :

167

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Générer plusieurs index

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage[makeindex]{imakeidx}

\makeindex[name = indexA, title = {Index A}]

\makeindex[name = indexB, title = {Index B}]

\usepackage[totoc, unbalanced]{idxlayout}

\begin{document}

Texte\index[indexA]{Texte} % Va dans l'index 1

\newpage

Autre essai \index[indexA]{Essai a confirmer@Essai à confirmer

} et j'ajouterai aussi \index[indexB]{Ajout!Ajout 1} que

les combinaisons sont possibles \index[indexB]{Ajout!Ajout

teste@Ajout testé}

\setindexprenote{Ceci est mon index !}

\printindex[indexA]{}

% \setindexprenote à réinitialiser (contrairement à

\indexprologue)

\setindexprenote{}

\printindex[indexB]{}

\end{document}

La petite filouterie

Enfin, il est possible d’utiliser un index pour classer des données (liste de

films, jeux, pistes musicales par nom d’auteur. . .). Au lieu de tenir un Excel

(tri facilité), tu peux renseigner tous ces éléments grâce à une entrée d’un

index et LATEX se charge de faire le tri.

168

Chapitre 12. Améliorer son texte et sa mise en forme

;A<

Il y a une toute petite astuce à connaître : supprimer le numéro de page.

En effet, ici, toutes nos entrées doivent être affichées sur une page bidon.

Dans le cas contraire, l’index ne sera pas généré (ou sera vide). Toutefois,

nous n’avons pas besoin des numéros de page. Il faut donc procéder ainsi :

Un index (bidon ?) sans les numéros de page

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage[makeindex]{imakeidx}

\makeindex[title = {Liste de films}]

\usepackage[totoc, unbalanced]{idxlayout}

\usepackage[colorlinks]{hyperref}

\begin{document}

Texte bidon\index{Film 1}\index{Film 2} \newpage

{

% Suppression du numéro de page dans l'index

\def\hyperpage#1{}

% Création de l'index

\printindex{}

}

\end{document}

Si tu essayes ce code, tu devrais te rendre compte qu’il fonctionne uniquement parce que le package hyperref est appelé, comme la commande

\hyperpage est redéfini localement comme étant vide.

De plus, cette solution permet de supprimer seulement le numéro de page.

Il reste encore la virgule de séparation, présente par défaut. Il faut donc

reprendre le fichier de personnalisation en .ist et l’adapter à ce nouveau

besoin.

169

Chapitre 13

Mathématiques : remarques &

astuces

J

e rédige très peu de formules mathématiques dans mon quotidien. Ne sois

donc pas surpris : si ce chapitre paraît étonnamment court, il contient

des informations précieuses !

13.1 Remarques générales

La commande officielle pour écrire un vecteur est \vec et non pas l’immonde \overrightarrow. Il est aussi possible d’utiliser la commande \vv du

package esvect, adaptée pour l’écriture de vecteurs.

%\usepackage{esvect} % Commande

\vv{AB}

Pour les vecteurs, utiliser $\vec

{u}$ ou $\vv{u}$ est mieux

que $\overrightarrow{u}$.

Pour les vecteurs, utiliser

⃗u ou #»u est mieux que −→u .

Pour placer des barres verticales, ne pas utiliser Alt Gr + 6 mais la commande \lvert pour la gauche ou \rvert pour la droite. \lVert et \rVert

sont aussi disponibles pour placer des doubles barres.

Dans le cas de délimiteurs, il faut donc procéder ainsi : \left\vert et

\right\vert ; \left\Vert et \right\Vert pour des doubles barres.

170

Chapitre 13. Mathématiques : remarques & astuces

;A<

Valeur absolue : $\lvert x \rvert

< 215$.

Norme : $\lVert \vec{u} \rVert$

ou $\left\Vert \frac{\vec{u

}}{13} \right\Vert$ (dé

limiteurs).

Valeur absolue : |x| <

215.

Norme : ∥⃗u∥ ou

⃗u

13

(délimiteurs).

Pour mettre des accolades en-dessous d’une formule en mode mathématiques, la commande underbrace est disponible. De même au-dessus avec

overbrace.

Un cas bidon : \[\underbrace{1 -

1 + 1 - 1}\_{= 0} + 13 = 13\]

De même : \[\overbrace{1 - 1 + 1

- 1}^{= 0} + 215 = 215\]

Un cas bidon :

1 − 1 + 1 − 1

| {z }

=0

+13 = 13

De même :

=0

z }| {

1 − 1 + 1 − 1 +215 = 215

Le package mathrsfs permet d’utiliser la commande \mathscr pour donner un style différent de celui fourni par \mathcal.

%\usepackage{mathrsfs} % Commande

\mathscr{C}

Changer la forme des lettres en

mode mathématiques est inté

ressant, comme avec $\mathcal

{X}$ pour le polynôme caracté

ristique ou $\mathscr{C}^0$

pour l'ensemble des fonctions

continues.

Changer la forme des

lettres en mode mathématiques est intéressant,

comme avec X pour le

polynôme caractéristique

ou C

0

pour l’ensemble

des fonctions continues.

Pour un intervalle avec des doubles barres, le package stmaryrd et les

commandes \llbracket et \rrbracket fonctionnent à merveille !

171

Chapitre 13. Mathématiques : remarques & astuces

;A<

%\usepackage{stmaryrd} %

Commandes \llbracket &

\rrbracket

Soit $n \in \mathbb{N}$. Soit $i

\in \llbracket 0 ; n

\rrbracket$.

Soit n ∈ N. Soit i ∈

J0; nK.

13.2 Limites et indiçage

Il existe d’autres cas plus techniques, comme l’écriture de limites ou de

plusieurs lignes d’indiçage dans une somme ou un produit :

❖ écriture d’une limite : combiner les commandes \underset, \to et

\lim ;

❖ empilement d’indices : emploi de \substack{<ind\_1> \\ <ind\_2>}.

Voyons sur des cas concrets comment nous servir de ces commandes :

Exemples concrets

Limite : \[\underset{x \to +

\infty}{\lim} f(x) = 0\]

Polynômes de Lagrange : \[L\_k =

\prod\_{\substack{j = 0 \\ j

\neq k}}^n \frac{X - a\_j}{a\_k

- a\_j}\]

Limite :

lim x→+∞

f(x) = 0

Polynômes de Lagrange :

Lk =

Yn

j=0

j̸=k

X − aj

ak − aj

La littérature montre l’existence d’une commande \limits qui peut servir

pour ces exemples. Non seulement j’ai lu qu’il était déconseillé de

l’utiliser 1

, mais tu peux obtenir un résultat propre sans cette commande

donc évitons de nous compliquer la vie !

1. Si tu veux savoir pourquoi, je te laisse chercher. Depuis le temps, j’ai oublié.

172

Chapitre 14

Tableaux & boîtes

I

l est possible d’obtenir des tableaux plus élaborés avec quelques commandes supplémentaires, sans parler des décorations avec des boîtes.

14.1 Autres formats de cellules

Dans la partie précédente de ce guide, j’ai présenté les formats par défaut

pour les cellules d’un tabular : l, c et r. Grâce au package array, il en

existe d’autres, qui permettent de prendre en compte une longueur comme

argument : les formats p{<longueur>}, m{<longueur>} et b{<longueur>}.

p (comme paragraph) permet de définir une colonne de largeur définie par

longueur et dont le contenu est aligné en haut à gauche ; m (comme middle)

aligne le contenu à gauche et le centre verticalement ; b (comme bottom)

aligne aussi le contenu à gauche mais le fixe en bas de la cellule.

Il n’est pas possible de cumuler des formats différents (limite du

package array vraisemblablement). Il ne faut donc pas s’étonner si le résultat ne correspond pas à tes attentes si tu définis une colonne au format p et

une autre au format m.

Si, comme nous le verrons par la suite, la commande linebreak est utile

pour écrire du texte sur une nouvelle ligne (dans la même cellule), elle peut

parfois créer de grandes espaces blancs entre les mots (justification du texte

forcée lors du retour à la ligne).

La meilleure solution, valable uniquement pour ces nouveaux formats, est

d’utiliser la commande \newline, qui permet donc d’éviter tous ces désagréments. Par exemple, cette solution peut se révéler pratique quand tu veux

renseigner plusieurs dates dans une même cellule :

173

Chapitre 14. Tableaux & boîtes

;A<

Le format de cellule p

% Ajout au préambule

% \usepackage{array}

\begin{tabular}{p{0.5\linewidth}|

p{0.3\linewidth}}

Du texte \linebreak Un très très

très long texte & {\hspace\*{

\fill}Titre centré\hspace\*{

\fill}} \\ \hline

Nouvel essai \newline Meilleur

espacement & OK

\end{tabular}

Du texte

Un très très

très long

texte

Titre

centré

Nouvel essai

Meilleur espacement

OK

Le format de cellule m

% Ajout au préambule

% \usepackage{array}

\begin{tabular}{m{0.3\linewidth}m

{0.5\linewidth}}

2018 \newline 2017 & Responsable

de projet (ligne de

production)

\end{tabular}

2018

2017

Responsable

de projet

(ligne de

production)

!

Nota Bene : centrage et retour à la ligne

Pour une raison que j’ignore, le centrage horizontal du texte dans

une cellule au format p / m / b ne fonctionne pas avec la commande

usuelle \hfill.

La meilleur solution consiste à utiliser la commande \hspace\*

combinée avec \fill de la manière suivante :

174

Chapitre 14. Tableaux & boîtes

;A<

!

\begin{tabular}{p{0.3\linewidth}}

{\hspace\*{\fill}Texte / Titre\hspace\*{\fill}} \\

Blablabla

\end{tabular}

Cette solution doit vraiment être utilisée de manière

ponctuelle ! Si tu veux la généraliser à toute la colonne, je te

recommande d’aller lire la section suivante.

Enfin, le retour forcé à la ligne avec \newline (ou \linebreak)

n’est bel et bien valide seulement pour les formats p / m / b.

14.2 Cellules centrées verticalement et horizontalement

Le format de colonne c ou m n’est pas suffisant pour centrer à lui tout seul

verticalement et horizontalement le contenu d’une cellule. Le plus simple est

d’utiliser le format m (centrage vertical) et de le combiner avec un descripteur

pour appeler la commande \centering (centrage horizontal).

La définition de la colonne dans l’environnement tabular se fait alors de

la manière suivante :

>{\centering\arraybackslash}m{<longueur\_case>}

Très clairement, le code devient illisible et pénible à reprendre si tu dois

copier-coller ce long format à chaque fois.

Grâce au package array, il est possible de créer son propre format de colonne grâce à la commande newcolumntype.

1 La syntaxe de cette commande

est la suivante :

Appel de la commande \newcolumntype

\newcolumntype{<nom\_format>}[<nbre\_arg>]{<def\_format>}

1. Je préconise même d’appeler cette commande dans le préambule, pour regrouper

toutes les commandes personnelles et y revenir plus facilement par la suite.

175

Chapitre 14. Tableaux & boîtes

;A<

Un exemple d’utilisation serait alors :

Un exemple avec un SWOT

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

% Ajout dans le préambule

\usepackage{graphicx} % Pour la commande rotatebox

\usepackage{array}

\newcolumntype{C}[1]{>{\centering\arraybackslash}m{#1}}

\begin{document}

\begin{tabular}{C{0.1\linewidth}\*{2}{|C{0.4\linewidth}}}

~ & {\Large{}Atouts} & {\Large{}Handicaps} \\ \hline

\rotatebox{90}{\Large{}Interne} & \textbf{Forces (Strengths)}

\begin{itemize}

\item point SA

\item point SB

\end{itemize} & \textbf{Faiblesses (Weaknesses)} \begin{

itemize}

\item point WA

\item point WB

\end{itemize} \\ \hline

\rotatebox{90}{\Large{}Marché} & \textbf{Opportunités (

Opportunities)} \begin{itemize}

\item point OA

\item point OB

\end{itemize} & \textbf{Menaces (Threats)} \begin{itemize}

\item point TA

\item point TB

\end{itemize}

\end{tabular}

% Ajuster les listes à puce selon ton bon vouloir

176

Chapitre 14. Tableaux & boîtes

;A<

\end{document}

Atouts Handicaps

Interne

Forces (Strengths)

— point SA

— point SB

Faiblesses (Weaknesses)

— point WA

— point WB

Marché

Opportunités

(Opportunities)

— point OA

— point OB

Menaces (Threats)

— point TA

— point TB

1

Le raccourci pour définir N colonnes identiques est toujours utilisable.

La formulation ne change pas : c’est toujours \*{<nbre\_col>}{<format>},

comme vu dans l’exemple précédent.

14.3 Fusion et coloriage de cellules

Si la fusion des colonnes et des lignes est comprise de base avec le package array, la fusion des lignes laisse un peu à désirer. C’est pourquoi il

vaut mieux utiliser le package multirow.

Pour fusionner des colonnes, il faut employer la commande \multicolumn,

avec la syntaxe suivante : \multicolumn{<nbre\_col>}{position}{texte}.

Pour avoir un séparateur adapté, une ligne partielle peut être insérée grâce

à la commande \cline{<col\_debut>-<col\_fin>}.

Pour la fusion des lignes, il faut utiliser la commande \multirow et la

syntaxe suivante : \multirow{<nbre\_ligne>}{\*}{Texte}. L’emploi de {\*}

permet d’avoir les cellules fusionnées avec la bonne largeur et d’éviter les

problèmes de centrage.

177

Chapitre 14. Tableaux & boîtes

;A<

Afin de pouvoir colorier les cellules, il faut ajouter le package xcolor dans

le préambule, avec l’option table. Grâce à la commande \rowcolor{<couleur>},

tu peux colorier une ligne entière.

\columncolor{<couleur>} et \cellcolor{<couleur>} font de même

respectivement pour une colonne et une cellule. Voyons tous ces éléments

dans un petit exemple :

Coloriage et fusion

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

% Ajout dans le préambule

\usepackage{array}

\usepackage{multirow} % Pour fusionner des lignes d'un tableau

\newcolumntype{C}[1]{>{\centering\arraybackslash}m{#1}}

\renewcommand{\arraystretch}{1.3} % Meilleure lisibilité

\usepackage[table]{xcolor} % [table] : pour colorer les

cellules (package colortbl)

\begin{document}

\begin{tabular}{|\*{4}{C{0.2\linewidth}|}}

\hline

\rowcolor{orange!80!yellow} \multicolumn{4}{|c|}{\textbf{

Comparaison des configurations}} \linebreak \\ \hline

\multirow{2}{\*}{\textbf{Critères}} & \multicolumn{3}{c|}{

\textbf{Structures}} \linebreak \\ \cline{2-4}

~& \textbf{Fonctionnelle} & \textbf{Divisionnelle} & \textbf{

Matricielle} \\ \hline \hline

Stabilité & ++ & + & - \\ \hline

Flexibilité \& Adaptabilité & - & - & ++ \\ \hline

\rowcolor{black} ~ & ~ & ~ & ~ \\ \hline

\textbf{Cas pratiques} & \multicolumn{3}{|c|}{\textbf{Qui ré

ussira le mieux à répondre aux besoins ?}} \\ \hline

178

Chapitre 14. Tableaux & boîtes

;A<

Projet (long) à réaliser & - & - & ++ \\ \hline

Fabrication en grande série & ++ & - & -- \\ \hline

\end{tabular}

\end{document}

\end{codedisplay}

Il se peut que la visualisation des bordures noires du tableau

ne s'affiche pas bien avec la cellule colorée. Il s'agit

juste d'un problème d'affichage avec ton écran d'

ordinateur, tellement la ligne est fine.

Sinon, tu peux aussi augmenter l'épaisseur des traits de ton

tableau, grâce à la commande \verb?\setlength

\arrayrulewidth{<épaisseur>}?.

\section{\texttt{longtable} \& \texttt{booktabs}}

J'ai découvert ces deux packages -- \verb?longtable? et \verb?

booktabs? -- en 2017, et depuis je ne m'en passe plus ! \\

Le premier se révèle très utile pour des tableaux dont la

longueur dépasse une page. Il suffit simplement de

remplacer l'environnement \verb?tabular? par \verb?

longtable? et tout le reste fonctionne de la même fa¸con, y

compris les formats de colonnes définis par l'utilisateur

ou la mise en place d'une légence (\verb?\caption?).

Tu peux aussi mettre une note en bas de page pour indiquer que

le tableau se poursuit en page suivante, rappeler lors du

changement de page les titres des colonnes ou encore

indiquer quand le tableau est terminé !

Bref, ce package offre pas mal de possibilités, que je te

laisse aller découvrir dans l'exemple qui suit et complé

ter le cas échéant grâce à la documentation officielle :

\url{https://www.ctan.org/pkg/longtable}. \\

Quant au second package, c'est le seul à ma connaissance qui

permette d'obtenir des tableaux de qualité, suffisamment a

érés, avec de bons séparateurs. En théorie, la commande

179

Chapitre 14. Tableaux & boîtes

;A<

\verb?\renewcommand{\arraystretch}{1.3}? permet de

modifier la hauteur de ligne, pour tous les tableaux créés

après cette commande.

Mais, avec des fractions, impossible de modifier quoi que ce

soit\dots{} sauf avec \verb?booktabs? ! Pourquoi ? Je n'en

sais rien. Le rendu est meilleur et convient.

\begin{coderesult}{Un exemple d'utilisation du package \verb?

longtable?}

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{lipsum}

\usepackage{array, longtable}

\renewcommand{\arraystretch}{1.3}

\begin{document}

\lipsum[1] \\

\begin{center}

\begin{longtable}{p{0.4\linewidth}p{0.3\linewidth}}

% Définition des headers & footers du longtable

{\hspace\*{\fill}\textbf{Colonne A}\hspace\*{\fill}} & {\hspace

\*{\fill}\textbf{Colonne B}\hspace\*{\fill}} \\ \hline

\endhead

\multicolumn{2}{r}{\textit{(suite sur la page suivante)}} \\

\endfoot

\multicolumn{2}{@{\hrulefill}c@{\hrulefill}}{\raisebox{-3pt

}{~~\textsc{Fin du tableau}~~}}

\endlastfoot

% Contenu du longtable

\lipsum[2] & \'Evident ! \\

\lipsum[3] & Limpide même !!!

180

Chapitre 14. Tableaux & boîtes

;A<

\end{longtable}

\end{center}

\end{document}

Comparaison des configurations

Critères Structures

Fonctionnelle Divisionnelle Matricielle

Stabilité ++ + - Flexibilité &

Adaptabilité - - ++

Cas

pratiques Qui réussira le mieux à répondre aux besoins ?

Projet (long) à

réaliser - - ++

Fabrication en

grande série ++ - –

1

Un exemple d’utilisation du package booktabs

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{amsmath, amsfonts, amssymb}

\newcommand{\module}[1]{\left\lvert #1 \right\rvert}

\usepackage{array, booktabs}

\renewcommand{\arraystretch}{1.3}

\newcolumntype{C}[1]{>{\centering\arraybackslash}m{#1}}

181

Chapitre 14. Tableaux & boîtes

;A<

\begin{document}

\everymath{\displaystyle}

\vspace\*{\fill}

\begin{center}

\begin{tabular}{\*{2}{C{0.3\linewidth}}} \toprule

$f(x)$ & $\int f(x)\,dx$ \\ \midrule % \midrule : commande

propre à booktabs - Création d'une ligne horizontale (=

\hline)

$x^\alpha$, avec $\alpha \neq -1$ & $\frac{x^{\alpha + 1}}{

\alpha + 1}$ \\ \midrule

$\frac{1}{x}$ & $\ln \module{x}$ \\ \midrule

$\cos (ax)$, avec $a \neq 0$ & $\frac{\sin (ax)}{a}$ \\

\midrule

$\sin x$ & $-\cos x$ \\ \midrule

$\frac{1}{1 + x^2}$ & $\arctan x$ \\ \midrule

$\cosh x$ & $\sinh x$ \\ \midrule

$\sinh x$ & $\cosh x$ \\ \midrule

$e^{\omega x}$, avec $\omega \neq 0$ & $\frac{e^{\omega x}}{

\omega}$ \\ \midrule

$\frac{u'}{u}$ & $\ln \module{u}$ \\ \midrule

$\tan x$ & $-\ln ( \module{\cos x})$ \\ \midrule

$\frac{1}{\sqrt{1 - x^2}}$ & $\arcsin x$ \\ \midrule

$\frac{-1}{\sqrt{1 - x^2}}$ & $\arccos x$ \\ \midrule

$\frac{1}{\sqrt{x^2 + 1}}$ & $\mathrm{argsinh}~x$ \\ \midrule

$\frac{1}{\sqrt{x^2 - 1}}$ & $\mathrm{argcosh}~x$ \\ \midrule

$\frac{1}{1 - x^2}$ & $\mathrm{argtanh}~x$

\end{tabular}

\end{center}

\vspace\*{\fill}

\end{document}

182

Chapitre 14. Tableaux & boîtes

;A<

f(x)

Z f(x) dx

x

α

, avec α ̸= −1 x

α+1

α + 1

1

x

ln |x|

cos(ax), avec a ̸= 0 sin(ax)

a

sin x − cos x

1

1 + x

2

arctan x

cosh x sinh x

sinh x cosh x

e

ωx, avec ω ̸= 0 e

ωx

ω

u

′

u

ln |u|

tan x − ln(|cos x|)

1

√

1 − x

2

arcsin x

−1

√

1 − x

2

arccos x

1

√

x

2 + 1

argsinh x

1

√

x

2 − 1

argcosh x

1

1 − x

2 argtanh x

1

Ici, j’ai séparé l’emploi de longtable et de booktabs pour bien distinguer

les deux exemples mais rien n’empêche d’utiliser l’environnement longtable

avec des commandes de booktabs (comme \toprule ou \midrule).

14.4 Créer sa propre boîte

Le package tcolorbox propose de très nombreux outils pour créer des

boîtes. Il propose même une commande pour créer soi-même son propre environnement, et donc sa propre boîte entièrement personnalisable.

La structure est très particulière et propre à tcolorbox mais permet de

respecter la philosophie LATEX : séparer le fond de la forme par l’intermédiaire

de commandes.

Un changement sur la commande entraîne alors un changement sur tout

le document. Tu admettras que c’est nettement plus pratique que de devoir

revenir sur tout le document et le corriger.

La création d’une nouvelle boîte repose sur une syntaxe identique à celle

utilisée lors de la création d’une nouvelle commande, avec [<nbre\_arg>]

absent s’il n’y a pas d’argument pour notre boîte :

183

Chapitre 14. Tableaux & boîtes

;A<

Création d’une nouvelle boîte sous tcolorbox

\newtcolorbox{<nom\_boite>}[<nbre\_arg>]{<options>}

Il est préconisé de définir toutes ses boîtes dans le préambule, là encore

pour faciliter les modifications futures et centraliser, regrouper toutes ces

commandes.

De plus, il est possible, comme pour la création de commandes, de définir

un argument optionnel selon la syntaxe suivante, avec l’argument optionnel

qui correspond automatiquement au choix #1 dans la définition de la commande :

Utilisation d’un argument optionnel

\nextcolorbox{<nom\_boite>}[<nbre\_arg>][<defaut>]{<options>}

Si <defaut> est présent, alors le premier argument spécifié par <nbre\_arg>

soit #1 est optionnel avec une valeur par défaut fixée à <defaut>.

Ainsi, si tu renseignes <defaut> par une valeur vide, ta boîte ne souffrira

pas d’un changement dans ses options, à moins de les notifier lors de son

appel ! Je pense qu’un premier exemple simple ne sera pas de trop pour

comprendre cette nouvelle notion :

Nouvelle commande & argument par défaut

\newcommand{\format}[2][\textbf]{

Je trouve que \LaTeX{} est

#1{#2 !}}

\format{merveilleux} \\

\format[\textit]{fantastique} \\

\format[\textsc]{incroyable}

Je trouve que LATEX est

merveilleux !

Je trouve que LATEX est

fantastique !

Je trouve que LATEX est

incroyable !

Nous avons donc le code complet suivant pour la création 2

, suivi de

quelques exemples. Attention à ne pas oublier d’appeler le package tcolorbox

2. Je ne vais pas revenir en détail sur toutes les options tcolorbox utilisées. Elles

sont décrites dans la documentation officielle, disponible sur https://www.ctan.org/

pkg/tcolorbox. À toi aussi de faire des essais !

184

Chapitre 14. Tableaux & boîtes

;A<

avec les options breakable et skins !

Ma petite boîte personnelle

\usepackage[breakable, skins]{tcolorbox}

% Boîte type générique

\newtcolorbox{boitetype}[4][]{enhanced, breakable, before

upper = {\parindent17.6pt}, beforeafter skip =

\baselineskip, colframe = #3, colback = #4, boxrule = 2pt,

arc = 4mm, fonttitle = \bfseries, title = {#2}, coltitle

= black, #1}

% La boîte utilisée

\newenvironment{boite}[3][]{\begin{boitetype}[#1]{#2}{#3}{

white}}{\end{boitetype}}

Application

\begin{boite}{Un premier exemple}{orange}

C'est pratique, n'est-ce pas ?

\end{boite}

\begin{boite}[colback = violet!50, coltitle = white]{Un deuxiè

me exemple}{violet}

Apportons quelques petits changements, juste pour cette fois.

\end{boite}

\begin{boite}{Un troisième exemple}{cyan}

Retour sur un cas normal d'utilisation.

\end{boite}

Un premier exemple

C’est pratique, n’est-ce pas ?

185

Chapitre 14. Tableaux & boîtes

;A<

Un deuxième exemple

Apportons quelques petits changements, juste pour cette

fois.

Un troisième exemple

Retour sur un cas normal d’utilisation.

Le principe est donc de créer une nouvelle boîte (syntaxe très similaire

à celle employée pour créer une commande) pour chaque cas (boîte pour les

définitions, boîte pour les remarques, etc.).

Pour t’éviter de devoir changer les options à chaque fois, le mieux reste

de créer une boîte type puis de créer tes différentes boîtes à partir de cette

boîte générique.

Et avec l’astuce de l’argument optionnel, tu laisses de la souplesse à ton

code, que tu peux donc adapter sur le pouce si tu as un cas exceptionnel à

traiter. Pratique et puissant à la fois !

14.5 Afficher du code LATEX

Il n’y a pas 13 façons d’afficher du code LATEX simplement 3

. La première

solution requiert d’utiliser l’environnement verbatim : tout ce qui est compris

dans cet environnement ne sera pas interprété par LATEX et sera recopié tel

quel.

Cependant, une trop longue commande dépasse des marges voire de la

page. . . C’est donc vite compliqué de lire la fin ou même de copier un morceau de code.

Toutefois, il existe déjà un moyen simple pour les “petites” formules :

la commande \verb?<cmde>?. Il faut juste écrire \verb et tout ce qui est

compris entre les deux délimiteurs (ici, des ?) subit le même traitement que

sous l’environnement verbatim (environnement verbatim local en quelque

sorte).

? n’est d’ailleurs pas le seul délimiteur envisageable : par exemple, !

convient tout à fait.

3. Écrire \ pour produire \ n’est vraiment pas une solution viable !

186

Chapitre 14. Tableaux & boîtes

;A<

Autrement, le seul package qui permette de mettre en forme du code

(LATEX, mais aussi Python, Perl, C, C++, Java, SQL, HTML, etc.) avec un

retour à la ligne intégré s’appelle listings.

Beaucoup d’options de mise en forme sont disponibles (numéro de ligne

de code sur le côté, commandes en couleur selon le langage. . .) mais il y a

mieux. Et oui, c’est maintenant que revient le Saint Graal : tcolorbox ! Ce

dernier intègre le package listings et permet de retourner un résultat soigné

et personnalisable.

Pour obtenir le code le plus propre et tenir compte des contraintes supplémentaires du package listings, il faut procéder en plusieurs étapes :

1) Charger le package tcolorbox avec les options breakable & skins,

pour la personnalisation des boîtes, ainsi que listings, pour intégrer

le package éponyme.

2) Intégrer les accents avec \lstset{literate = <accents>}. Le package listings n’interprète pas nativement les accents donc il faut

les définir au préalable.

La programmation se fait généralement en anglais donc ne comporte

pas d’accents. Mais personnellement, en tant que Français qui rédige

un guide en français, mes commentaires contiennent parfois des accents

donc il faut mettre en place cette dernière astuce pour s’en sortir.

3) Définir un style pour l’affichage de la boîte, qui contiendra toutes les

options liées à sa personnalisation.

En effet, contrairement à la section précédente où nous avons défini une

boîte type puis des dépendances, l’ajout d’une couche supplémentaire

avec le package listings nuit à sa création et entraîne des erreurs

irrémédiables actuellement.

Il faut donc ruser et créer autant d’environnements tcolorbox avec

listings que de boîtes différentes pour afficher le code, mais qui vont

à chaque fois appeler les styles mis en place. Les définitions deviennent

concises et les boîtes peuvent se généraliser facilement.

4) Créer un style listings, pour la personnalisation du code à proprement parler.

5) Créer la boîte (environnement tcolorbox).

187

Chapitre 14. Tableaux & boîtes

;A<

Voyons directement avec un exemple les points précédemment abordés,

pour découvrir les commandes à utiliser et leur fonctionnement :

Écrire du code sous LATEX : paramétrage

% Chargement de tcolorbox et des options nécessaires

\usepackage[breakable, listings, skins]{tcolorbox}

% Pour résoudre le problème des accents dans le code (listings

sous tcolorbox)

\lstset{literate = {à}{{\`a}}1 {â}{{\^a}}1 {é}{{\'e}}1 {è}{{\`

e}}1 {ê}{{\^e}}1 {î}{{\^i}}1 {ô}{{\^o}}1 {ù}{{\`u}}1 {û}{{

\^u}}1 {¸c}{{\c{}c}}1}

% Définition du style de la boîte (affichage)

\tcbset{

codemainoptions/.style = {

maincolor/.store in = {\tcbmaincol},

maincolor = LimeGreen,

rulewidth/.store in = {\tcbrulewidth},

rulewidth = 2pt,

enhanced, breakable, beforeafter skip = \baselineskip,

sharp corners, boxrule = \tcbrulewidth, colframe =

\tcbmaincol, colback = \tcbmaincol!15, drop fuzzy shadow,

colbacktitle = \tcbmaincol!50, coltitle = black, fonttitle

= \bfseries, title = {#1},

attach boxed title to top center = {yshift = -

\tcbrulewidth/2-\tcboxedtitleheight/2, yshifttext = -

\tcboxedtitleheight/2}, boxed title style = {boxrule =

\tcbrulewidth, frame code = {

\path[tcb fill frame] ([xshift = -3mm]frame.west) -- (

frame.north west) -- (frame.north east) -- ([xshift = 3mm]

frame.east) -- (frame.south east) -- (frame.south west) --

cycle;}, interior code = {

\path[tcb fill interior] ([xshift = -2mm]interior.west

) -- (interior.north west) -- (interior.north east) -- ([

xshift = 2mm]interior.east) -- (interior.south east) -- (

interior.south west) -- cycle;}}

}

}

% Création du style listings

188

Chapitre 14. Tableaux & boîtes

;A<

\lstdefinestyle{mainlststyle}{

language = {[LaTeX]TeX},

style = tcblatex,

texcsstyle = \*\color{cyan!65!black},

commentstyle = \color{gray},

tabsize = 4,

keepspaces = true,

breaklines = true,

breakatwhitespace = false,

inputencoding = utf8,

numbers = none,

showspaces = false,

showtabs = false,

showstringspaces = false

}

% Création des boîtes "tcolorbox + listings" (environnement)

\newtcblisting{code}[2][]{codemainoptions = {#2}, listing

options = {style = mainlststyle}, listing only, #1}

\newtcblisting{codedisplay}[2][]{codemainoptions = {#2},

listing options = {style = mainlststyle}, listing side

text, righthand ratio = 0.4, sidebyside gap = 13mm,

bicolor, colbacklower = white, #1}

% N.B. : #2 = titre de la boite

% N.B. : option "breakable" pas bien compatible avec "listing

side text" ==> faire des "petits bouts" de code dans ce

cas

Application

\begin{verbatim}

Très pratique d'écrire du code sous \LaTeX{} \\

Mais c'est plus compliqué quand le code en question est trop

long !

\end{verbatim}

189

Chapitre 14. Tableaux & boîtes

;A<

\begin{code}{Boîte}

Très pratique désormais d'écrire du code sous \LaTeX{} !

Surtout si le code est extrêmement long.

\end{code}

\`A toi d'essayer l'environnement \verb?codedisplay? !

Très pratique d'écrire du code sous \LaTeX{} \\

Mais c'est plus compliqué quand le code en question est trop long !

Boîte

Très pratique désormais d'écrire du code sous \LaTeX{} !

Surtout si le code est extrêmement long.

À toi d’essayer l’environnement codedisplay !

Il est possible d’aller encore plus loin, toujours grâce aux packages tcolorbox

et listings, et de compléter le précédent code pour avoir les options suivantes : numérotation automatique des boîtes et création d’un sommaire

dédié à ces boîtes.

La documentation tcolorbox fournit toutes les indications nécessaires.

Les commandes nécessaires à utiliser sont décrites ci-après :

Intégrer du code – Aller encore plus loin

% Appel de tcolorbox, \lstset & création des styles

\usepackage[]{hyperref} % Obligatoire

% Reprise du style codemainoptions + title = {Code

\thetcbcounter{} : #1}

% Ajout : list entry = {\protect\numberline{\thetcbcounter}#1}

% Création d'une boîte numérotée pour le code

\newtcblisting[auto counter, number within = chapter, list

190

Chapitre 14. Tableaux & boîtes

;A<

inside = LaTeXcode]{code}[2][]{codemainoptions = {#2},

listing options = {style = mainlststyle}, listing only,

#1}

\begin{document}

% Ajout du nouveau sommaire (hyperref)

\phantomsection

\addcontentsline{toc}{chapter}{Liste des codes \LaTeX{}} %

Ajout dans le sommaire

\tcblistof[\chapter\*]{LaTeXcode}{Liste des codes \LaTeX{}} %

Sommaire dédié tcolorbox (LaTeXcode = référence)

Rédaction du document et utilisation de l'environnement \verb?

code? !

\end{document}

Enfin, il existe un autre package que listings, qui peut s’occuper automatiquement du coloriage du code : minted. Il fonctionne grâce à pygments,

une bibliothèque Python. Par contre, même après son installation, je n’ai

toujours pas réussi à le faire fonctionner. . .

191

Chapitre 15

Images : de nouvelles subtilités

C

omme pour les mathématiques, l’insertion d’images sous LATEX est réalisée sans surprise et sans avoir à connaître des techniques très poussées.

Cependant, au fur et à mesure de la rédaction de tes documents, tu constateras que tu souhaiteras améliorer certains détails. Voici donc quelques astuces supplémentaires toujours utiles !

15.1 Une référence toute prête

Les références, c’est bien. Les automatiser, c’est mieux. J’étais plutôt

agacé d’écrire constamment « (cf. Figure <ref> p. <page-ref>) », d’autant

plus que le mot « Figure » peut varier selon la classe.

Puis, j’ai découvert la commande \figurename{} : elle contient justement le nom utilisé dans la légende. Il est donc possible d’automatiser mon

problème initial grâce à une commande.

Mais c’est sans compter sur le package hyperref qui propose déjà une

commande toute prête à ce sujet : \autoref{<label>}, qui écrit directement

« Figure <ref> ».

Mais, si comme moi tu es un puriste et tu tiens à reprendre l’intitulé

exact de la légende (soit « Figure », en petites capitales), il faut procéder

à un petit correctif manuel après le préambule (pour éviter les conflits avec

le package babel) :

\renewcommand{\figureautorefname}{\figurename{}}

\renewcommand{\tableautorefname}{\tablename{}}

192

Chapitre 15. Images : de nouvelles subtilités

;A<

Et voilà, c’est tout ! Tu peux désormais t’amuser avec cette nouvelle commande très pratique.

15.2 Insérer des légendes intermédiaires

Il peut être intéressant d’afficher plusieurs images avec chacune sa légende,

ainsi qu’une légende globale pour toutes les images. Une solution très simple

est possible grâce au package subcaption et de la commande éponyme :

Utilisation de subcaption

% Ajout dans le préambule

%\usepackage{graphicx, float,

subcaption}

\begin{figure}[H]

\begin{minipage}{0.45\linewidth}

\includegraphics[width =

\linewidth]{fond.jpg}

\subcaption{Image A}

\end{minipage}

\hfill

\begin{minipage}{0.45\linewidth}

\includegraphics[width =

\linewidth]{fond.jpg}

\subcaption{Image B}

\end{minipage}

\caption{Images A \& B}

\end{figure}

(a) Image A (b) Image B

Figure 15.1 – Images A

& B

Les références continuent de fonctionner sans contrainte supplémentaire.

Ne pas hésiter à faire des tests le cas échéant.

!

subfigure : l’environnement à bannir ! ! !

Le package subcaption met à disposition un nouvel environnement, subfigure, qui présente peu d’intérêt et dont l’utilisation est

à bannir a

Pour rappel/précision, les commandes \caption ou \subcaption

193

Chapitre 15. Images : de nouvelles subtilités

;A<

!

ne sont valides que dans un élément flottant soit dans un environnement global figure.

a. Résultat d’une recherche sur un forum un jour, dont je n’ai plus la source.

Enfin, le package subcaption charge aussi le package caption, qui permet d’utiliser la commande \caption\*. Cette dernière permet d’avoir une

légende sans numéro. C’est toujours pratique de temps en temps.

15.3 Insérer un grand nombre de fichiers

Il est possible d’être amené, ponctuellement, à regrouper un grande nombre

de fichiers (images, PDF. . .) dans un seul et unique PDF.

Si écrire toutes les lignes de code ou faire des copier-coller pour n’avoir

qu’à modifier les noms de fichiers à la fin peut fonctionner, il existe une

méthode plus élégante et efficace qui consiste à utiliser une boucle for sous

LATEX. Tout est résumé dans le code ci-après :

Insertion avec une boucle for

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{graphicx, float} % Si images

\graphicspath{{./Images/}} % Chemin des images

\DeclareGraphicsExtensions{.jpg} % Pour définir l'extension

des images

\usepackage{pgffor} % Pour la boucle for

\begin{document}

% Page de garde ou ce que tu veux

% Commande d'insertion avec la boucle for

194

Chapitre 15. Images : de nouvelles subtilités

;A<

% #1 = numéro début

% #2 = numéro fin

% #3 = nom devant le numéro

\newcommand\*{\insertgraphicsfiles}[3]{%

\foreach \i in {#1,...,#2} {%

\vspace\*{\fill}

\begin{figure}[H]

\centering

\includegraphics[width = 0.99\linewidth]{#3\i}

\end{figure}

\vspace\*{\fill}

}

}

% Insertion images 001 à 009

\insertgraphicsfiles{1}{9}{00}

% Insertion images 010 à 099

\insertgraphicsfiles{10}{99}{0}

% Insertion images 100 à 151

\insertgraphicsfiles{100}{151}{}

% Il est possible de faire de même avec des PDF et \includepdf

\end{document}

Et voilà ! Le code peut paraître un peu saugrenu car j’ai choisi de numéroter les images de 001 à 999, pour garantir le bon rangement par ordre

alphabétique dans l’ordinateur.

Tu peux bien entendu simplifier le code présenté ou l’adapter selon la

façon dont tu nommes tes images.

Si tu ne juges pas ce passage intéressant, la réalisation d’un trombinoscope

peut constituer une application plus concrète de l’utilisation d’une boucle

for :

195

Chapitre 15. Images : de nouvelles subtilités

;A<

Réaliser un trombinoscope

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{graphicx}

\graphicspath{{./Images/}} % Chemin des images

\DeclareGraphicsExtensions{.jpg} % Pour définir l'extension

des images

\usepackage{array}

\usepackage{pgffor} % Pour les boucles

\setlength\parindent{0pt} % Pour supprimer les indentations (

inutiles ici)

\newcommand\*{\affiche}[3]{\begin{tabular}{c} \includegraphics[

width = 0.32\linewidth]{#1} \\ #2 \textsc{#3} \end{tabular

} }

% Attention, l'espace après le \end{tabular} est indispensable

pour les renvois. Sinon, tout s'affiche sur une seule

ligne

\begin{document}

\begin{center}

\huge{\textsc{Titre}}

\end{center}

% Application avec mon image fond.jpg

\foreach \prenom/\nom/\fichier in {%

Prénom/Nom/fond,%

Prénom/Nom/fond,%

Prénom/Nom/fond,%

Prénom/Nom/fond} {\affiche{\fichier}{\prenom}{\nom}}

% Et ainsi de suite. Il vaut mieux ne pas mettre d'accent ni d

196

Chapitre 15. Images : de nouvelles subtilités

;A<

'espace dans les noms de fichiers

% Le % en fin de ligne est indispensable au bon fonctionnement

de \foreach pour éviter l'insertion de blancs qui

troubleraient l'appel du fichier (et permet de faciliter

la relecture du code)

\end{document}

Titre

Prénom Nom Prénom Nom Prénom Nom

Prénom Nom

1

Nous reviendrons plus tard, avec le chapitre sur TikZ, chapitre 17 p. 215,

sur d’autres utilisations de la boucle for sous LATEX.

15.4 Insérer un fichier .svg

Si, comme moi, tu apprécies ne pas avoir de gros pixels immondes au

moindre zoom de ton fichier PDF, il est possible d’importer un fichier .svg

(image vectorielle donc pas de pixels au zoom) dans ton document.

Pour ce faire, aucun package supplémentaire n’est requis et il faut juste

suivre la procédure suivante :

197

Chapitre 15. Images : de nouvelles subtilités

;A<

1) Enregistrer le fichier .svg sous Inkscape au format .pdf (option Enregistrer sous).

2) Dans les options, choisir “Exclure le texte. . .” et “Utiliser la taille. . .”.

3) Garder les deux fichiers générés (.pdf et .pdf\\_tex).

4) Utiliser le code ci-après et compiler le tout.

Insérer un fichier .svg

\begin{figure}[H]

\centering

\def\svgwidth{\columnwidth} % Pour définir la largeur de l'

image

%\input{<nom\_fichier>.pdf\_tex}

\caption{Légende éventuelle}

\end{figure}

% \def\svgwidth{0.8\linewidth} est aussi envisageable

Et voilà, c’est tout ce qu’il y a à faire. Après, c’est vraiment se prendre le

chou pour pas grand chose. Autant rester sous Inkscape, enregistrer l’image

au format .eps et l’intégrer comme n’importe quelle image.

Les pixels ne se verront toujours pas au zoom et la compilation se fait

sans souci sous PDFLATEX (création d’un fichier intermédiaire supplémentaire mais génération bien plus rapide).

Bref, c’était surtout une volonté personnelle d’explorer de nouveaux domaines sous LATEX mais il faut aussi savoir utiliser des solutions simples

parfois.

198

Chapitre 16

Dessiner avec PSTricks

I

l n’y a rien de pire que d’apprendre une notion, de l’appréhender, d’expérimenter. . . pour se rendre compte qu’une autre est meilleure et qu’il

faille tout recommencer depuis le début.

C’est ce qui m’est arrivé avec PSTricks. J’ai appris à dessiner avec ce package, qui requiert de compiler avec le moteur LATEX ou XeLATEX. Puis, j’ai

découvert TikZ, qui fonctionne avec n’importe quel moteur de compilation.

Je ne vais pas supprimer mon travail initial. Tu peux le consulter. La

dernière mise à jour de ce chapitre date du 26 février 2019. Dans tous les

cas, je te recommande de passer directement au chapitre suivant

sur TikZ.

16.1 Fonctionnement général

Selon le dessin à réaliser, il faut charger un ou plusieurs packages :

→ pstricks : la base pour dessiner avec PSTricks ;

→ pst-circ : pour dessiner des circuits électriques ;

→ pst-node : pour dessiner des diagrammes ;

→ pst-eucl : pour dessiner des figures géométriques ;

→ pstricks-add : pour ajouter de nouvelles commandes, comme la rotation d’objets par exemple.

199

Chapitre 16. Dessiner avec PSTricks

;A<

Ensuite, pour indiquer à LATEX que nous souhaitons dessiner une image

avec PSTricks, il faut utiliser l’environnement pspicture, suivi de la taille

maximale de l’image au format (x\_max,y\_max).

Une option supplémentaire, [showgrid = true], est très utile pour visualiser le résultat avec un quadrillage en arrière-plan. Ce dernier permet de

corriger des points mal placés ou de faciliter les décalages à faire.

Bon, allons faire quelques essais pour mieux saisir le principe de fonctionnement.

16.2 Dessiner des circuits électriques

Le principe de fonctionnement est très simple. Imagine que tu dessines

ton circuit électrique sur une feuille de papier. Dans le coin inférieur gauche,

tu places un repère et son origine puis tu considères qu’un composant, un fil,

etc. revient à se déplacer d’une unité.

Honnêtement, si tu es arrivé jusqu’à cette partie du guide, tu devrais

pouvoir aller jeter un coup d’œil à l’aide du package sans problème, surtout

pour avoir accès à toutes les options disponibles. Voici deux petits exemples

pour te mettre en bouche :

Un cas minimaliste

\documentclass[a4paper, 12pt]{report}

% LaTeX // XeLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{amsmath, amsfonts, amssymb}

\usepackage[dvipsnames]{xcolor} % Pour les couleurs si besoin

\usepackage{pst-circ} % Pour les circuits électriques

\begin{document}

\everymath{\displaystyle}

200

Chapitre 16. Dessiner avec PSTricks

;A<

\begin{pspicture}[showgrid = true](5,2)

% showgrid affiche le quadrillage

% Permet de se repérer au début et en cas d'erreur

% A mettre sur false lors de la génération du résultat final

% Composants

\resistor(1,1)(2,1){$R$}

% Les coordonnées à renseigner sont celles des extrémités du

composant

\coil[dipolestyle = curved](3,1)(4,1){$L$}

% Fils

\wire[intensitylabel = $I$, intensitycolor = red,

intensitylabelcolor = red](0,1)(1,1)

\wire(2,1)(3,1)

\wire(4,1)(5,1)

% Annotations

\tension[labeloffset = -0.5](0.5,0.5)(2.5,0.5){$V$}

% Si coordonnées non entières, utiliser un point

\end{pspicture}

\end{document}

Un cas plus complet

\documentclass[a4paper, 12pt]{report}

% LaTeX // XeLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{amsmath, amsfonts, amssymb}

\usepackage[dvipsnames]{xcolor} % Pour les couleurs si besoin

\usepackage{pst-circ} % Pour les circuits électriques

201

Chapitre 16. Dessiner avec PSTricks

;A<

\begin{document}

\everymath{\displaystyle}

\begin{pspicture}[showgrid = true](8,3)

% Composants

\resistor(2,1)(2,2){$R$}

\coil[dipolestyle = curved](4,1)(4,2){$L$} % Un affichage

possible pour une bobine

\coil[dipolestyle = elektor](6,3)(7,3){$l$} % Un autre format

d'affichage

\resistor(8,1)(8,2){$\frac{r}{g}$}

% Fils

\wire[intensitylabel = $I$, intensitylabeloffset = 0.5](0,3)

(2,3)

\wire(2,3)(4,3)

\wire[intensitylabel = $I'$](4,3)(6,3)

\wire(7,3)(8,3)

\wire(0,0)(8,0)

\wire(2,0)(2,1)

\wire(2,2)(2,3)

\wire(4,0)(4,1)

\wire(4,2)(4,3)

\wire(8,0)(8,1)

\wire(8,2)(8,3)

% Annotations

\tension(0,0)(0,3){$V$}

\end{pspicture}

\end{document}

Conseil personnel

La génération sous XeLATEX peut se révéler assez longue, surtout

si tu cumules de nombreux circuits.

Après des essais, le temps d’attente est négligeable avec une compilation sous LATEX, suivie des conversions d’usage Dvi -> PS puis

202

Chapitre 16. Dessiner avec PSTricks

;A<

PS -> PDF.

Tu peux donc éventuellement rédiger tout ton rapport avec ce dernier moteur de compilation. Pour rappel, ce dernier ne tolère pas les

fichiers .png ou .jpg pour les images. Il faut donc les convertir en

fichier .eps, grâce au logiciel GIMP par exemple.

!

Nota Bene

Tu as peut-être déjà remarqué que le guide de pst-circ utilise

une commande \pnode pour définir les nœuds et leur donner une

lettre.

Il ne s’agit en aucun d’une obligation, comme l’attestent mes précédents exemples. Personnellement, pour un petit schéma, je ne recommande pas de le faire. Je trouve que c’est plus beaucoup plus long

s’il faut déplacer des points.

Cette solution se révèle toutefois plus pratique pour de grands

schémas, s’il y a beaucoup de changements à réaliser avant d’obtenir

le résultat souhaité et si tu as beaucoup de points communs, par

exemple.

Bon, si tu viens de te rendre compte que LATEX est extrêmement puissant

pour dessiner des circuits d’aussi bonne qualité, sache que ce n’est pas fini.

Allons dessiner tout court.

16.3 Dessiner tout court

Pour dessiner avec PSTricks, le principe est extrêmement similaire : tu

définis des traits ou des formes à partir de coordonnées et LATEX trace le tout.

C’est parti avec un exemple :

Un premier dessin : transmission de la chaleur

\documentclass[a4paper, 12pt]{report}

% LaTeX // XeLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

203

Chapitre 16. Dessiner avec PSTricks

;A<

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{amsmath, amsfonts, amssymb}

\usepackage[dvipsnames]{xcolor}

\usepackage{pstricks}

% Pas besoin du package xcolor ici

% pstricks l'importe automatiquement

\begin{document}

% Coefficient de transmission thermique d'une paroi

\begin{pspicture}(7,4)

\psline[linecolor = Green](1.5,4)(1.5,0) % Pour tracer une

ligne

\psline(2.5,4)(2.5,0)

\psline(3.5,4)(3.5,0)

\psline(4.5,4)(4.5,0)

\psline[linecolor = Green](5.5,4)(5.5,0)

\psline[linecolor = red]{->}(0,2)(7,2)

\psframe[fillstyle = hlines](1.5,0)(2.5,4) % Pour tracer un

rectangle

\psframe[fillstyle = vlines](3.5,0)(4.5,4)

\psframe[fillstyle = crosshatch](4.5,0)(5.5,4)

\rput(2,-0.25){1} % Pour placer une information

\rput(3,-0.25){2}

\rput(4,-0.25){3}

\rput(5,-0.25){4}

\rput(1.5,4.25){\textcolor{Green}{$T\_{S\_a}$}}

\rput(5.5,4.25){\textcolor{Green}{$T\_{S\_b}$}}

\rput(7,1.75){\textcolor{red}{$\Phi$}}

\rput(0,3){Ambiance a}

\rput(7,3){Ambiance b}

\rput(0,2.5){$T\_a$}

\rput(7,2.5){$T\_b$}

\end{pspicture}

204

Chapitre 16. Dessiner avec PSTricks

;A<

\end{document}

Un second dessin : tracé et hachurage

\documentclass[a4paper, 12pt]{report}

% LaTeX // XeLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{amsmath, amsfonts, amssymb}

\usepackage{pstricks}

% Pas besoin du package xcolor ici

% pstricks l'importe automatiquement

\begin{document}

% Création d'un domaine hachuré

\begin{pspicture}(7,7)

% Repère

\psline{->}(0,1)(7,1)

\psline{->}(1,0)(1,7)

\uput[d](7,1){$t\_A$} % Plus pratique pour placer une

indication décalée

% d = down ; u = up ; l = left ; r = right

\uput[ul](1,7){$t\_B$} % Combinaison de position possible DANS

CET ORDRE (lu ne fonctionne pas)

\pscircle[fillcolor = black, fillstyle = solid](6,1){0.1} %

Pour tracer un cercle + le remplir

\uput[d](6,1){30}

% Carré et Delta\_t (domaine hachuré)

\psline(6,1)(6,6)(1,6)

\pspolygon[linecolor = red, hatchcolor = red, fillstyle =

hlines](3,1)(6,4)(6,6)(4,6)(1,3)(1,1)(3,1)

205

Chapitre 16. Dessiner avec PSTricks

;A<

\rput(6.4,6.4){\textcolor{red}{$\Delta\_t$}}

\pscircle[linecolor = red, fillcolor = red, fillstyle = solid

](3,1){0.1}

\rput(3,0.6){\textcolor{red}{$t$}}

\pscircle[linecolor = red, fillcolor = red, fillstyle = solid

](1,3){0.1}

\rput(0.6,3){\textcolor{red}{$t$}}

\end{pspicture}

\end{document}

Pour plus de commandes

Je ne vois aucun intérêt à faire une liste des commandes et des

options possibles. Je t’ai fourni deux exemples pour que tu aies un

aperçu du rendu et des possibilités mais à toi d’aller te documenter

par la suite.

Je te recommande particulièrement d’aller sur : http:

//fr.wikibooks.org/wiki/LaTeX/Dessiner\_avec\_LaTeX/

Dessiner\_avec\_PSTricks. C’est assez complet.

16.4 Utiliser des coordonnées

Dans une optique d’automatisation des dessins (un système d’amortisseur

en mécanique ou un circuit RLC, utilisés de nombreuses fois, par exemple),

il faudrait pouvoir créer une commande.

L’argument principal de cette commande serait alors un point de départ

pour le schéma (en bas à gauche, en haut à droite ou ailleurs, au choix). Sous

PSTricks, il s’agirait d’un nœud (node) et tous les autres sont définis à partir

de ce nœud d’origine (décalage des abscisses et des ordonnées).

L’origine sert donc de “point d’ancrage” pour positionner le dessin et le

reste est construit automatiquement. Pour ce faire, il faut procéder de la

manière suivante :

❖ en plus de pstricks, charger le package pst-node ;

206

Chapitre 16. Dessiner avec PSTricks

;A<

❖ définir tous les nœuds grâce à la commande :

\psnodes(x1,y1){noeud1}..(xN,yN){noeudN}

En l’occurrence, le nœud 1 est l’origine ; (x1,y1) est donc remplacé

par (#1) (argument de la commande) ;

❖ définir les (xi,yi) en commençant par un ! et selon la méthode NPI

(cf. encadré ci-après) ;

❖ récupérer les coordonnées selon l’une des deux manières suivantes :

→ utiliser la commande \psGetNodeCenter{noeudi} noeudi.Z, où

Z correspond à x ou y (respectivement, récupération de l’abscisse

ou de l’ordonnée),

→ ou bien, introduire la commande saveNodeCoors dans les options de l’environnement pspicture et utiliser ensuite la syntaxe

N-noeudi.Z.

Il est aussi possible de définir des longueurs pour continuer de généraliser

la commande, comme nous le verrons dans l’exemple qui va suivre.

La Notation Polonaise Inverse

«

La notation polonaise inverse (NPI) (en anglais RPN pour Reverse

Polish Notation), également connue sous le nom de notation post-fixée,

permet d’écrire de façon non ambiguë les formules arithmétiques sans

utiliser de parenthèses.

»

Wikipédia – Disponible sur :

https://fr.wikipedia.org/wiki/Notation\_polonaise\_inverse

Concrètement, pour utiliser un exemple, l’opération ((1+2)×4)+3

peut être notée en NPI 1 2 + 4 x 3 +. Il suffit de partir de la gauche,

de prendre deux éléments et un opérateur, de faire le calcul et de le

remplacer. Pour détailler, nous avons donc ici :

→ 1 2 + 4 x 3 + : prendre 1 2 + qui devient 1 + 2 soit 3 ;

→ passage à 3 4 x 3 + : prendre 3 4 x qui devient 3 × 4 soit 12 ;

207

Chapitre 16. Dessiner avec PSTricks

;A<

→ passage à 12 3 + qui devient 12 + 3 soit 15.

Dans le cadre de PSTricks, le fonctionnement est le même sauf que

les opérateurs suivants sont utilisés : add, sub, mul et div, respectivement pour addition, soustraction, multiplication et division.

Avec des exemples commentés, nous obtenons des cas d’utilisation possibles :

Exemple abstrait (boîte)

% Compiler avec le moteur LaTeX

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{amsmath, amsfonts, amssymb}

\usepackage[dvipsnames]{xcolor}

\usepackage{pstricks, pst-node, pstricks-add}

% pst-node pour les noeuds et le calcul de nouvelles coordonné

es

% pstricks-add pour la commande \psrotate

\newcommand{\textedbox}[4]{\pnodes(#1){origine}(#2){fin}

\psframe[#3](origine)(fin)

\rput(!N-fin.x N-origine.x add 2 div N-fin.y N-origine.y add 2

div){\parbox{\linewidth}{\centering{}#4}}}

\begin{document}

\begin{pspicture}[showgrid = true, saveNodeCoors](10,5)

% saveNodeCoors ssi utilisation de N-node\_name.x/y

\def\longueur{4 } % Espace OBLIGATOIRE (sinon rien ne s'

affiche)

\def\decalage{0.5 }

208

Chapitre 16. Dessiner avec PSTricks

;A<

% Définition de longueurs

% Possibilité de les mettre en argument d'une commande

% Une option brute

\pnodes(1,1){origine}(!\psGetNodeCenter{origine} origine.x

\longueur add origine.y \longueur add){fin}

\psframe(origine)(fin)

\psline[linecolor = violet]{|<->|}(!N-origine.x N-origine.y

\decalage sub)(!N-fin.x N-fin.y \longueur \decalage add sub)

% NE PAS écrire \longueur{}...

% Une commande créée avec l'option saveNodeCoors

\rput(4,0){\psrotate(2.5,2.5){90}{\textedbox{0,2}{5,3}{

linecolor = red, framearc = 0.5, linestyle = dashed,

fillstyle = hlines, hatchcolor = gray}{\textcolor{cyan}{

Texte}}}}

% Une autre possibilité

\rput(9,2.5){\psframebox[linecolor = orange, framesep = 13pt]{

\Large{}Test}}

\end{pspicture}

\end{document}

Exemple concret (amortisseur)

% Compiler avec le moteur LaTeX

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{amsmath, amsfonts, amssymb}

\usepackage[dvipsnames]{xcolor}

\usepackage{pstricks, pst-node}

% pst-node pour les noeuds et le calcul de nouvelles coordonné

209

Chapitre 16. Dessiner avec PSTricks

;A<

es

\newcommand{\amortisseur}[3]{\pnodes(#1,#2){A}(!#1 1 sub #2){B

}(!#1 1 sub #2 1 add){C}(!#1 #2 1 add){D}(!#1 0.5 sub #2 1

add){F}(!#1 0.5 sub #2){E}(!#1 0.5 sub #2 0.5 add){G}(!#1

0.5 add #2 0.5 add){H}(!#1 1 sub #2 0.5 add){I}(!#1 2 sub

#2 0.5 add){J}(!#1 0.5 sub #2){K}\psline(A)(B)(C)(D)

\psline(F)(E) \psline(G)(H) \psline(I)(J) \uput[d](K){#3}}

\begin{document}

Un cas plus concret avec un amortisseur (taille fixe), moins é

légante mais qui fonctionne :

\begin{pspicture}[showgrid = true](3,2)

\amortisseur{2}{0.5}{$\mu$}

\end{pspicture}

\end{document}

Comme tu peux le constater, la définition des nœuds avec cette méthode

est, certes, laborieuse mais peut se révéler très pratique avec la possibilité de

créer des commandes : au lieu d’avoir une entrée pour l’abscisse de l’origine

et une autre pour son ordonnée, tout passe avec un argument et PSTricks

fait le reste.

Autrement, dans la définition des nœuds, avec cette notation, il ne faut

pas oublier le ! et il est important de noter que la séparation des abscisses

et des ordonnées se fait SANS virgule 1

.

16.5 Des boîtes pour le texte

Peut-être l’as-tu remarqué dans mon précédent exemple abstrait mais il

est possible de créer des boîtes avec le texte centré, et plein d’autres options.

Ma commande, définie dans l’exemple précédent, serait “parfaite” (de

mon point de vue) s’il était possible d’extraire la longueur de la boîte pour

l’intégrer comme argument de la parbox. Sans succès pour l’instant.

1. Pourquoi ? Je n’en sais rien, ça marche comme ça et c’est très bien. Mais il doit bien

y avoir une raison. . .

210

Chapitre 16. Dessiner avec PSTricks

;A<

Mais il semblerait qu’elle fonctionne grâce à un petit \linewidth. Tant

mieux.

Sinon, il existe d’autres possibilités sous PSTricks comme la commande

\PSTextFrame. Une piste à explorer !

16.6 Réaliser des intersections

Tu as envie de tracer un contour qui correspond à l’intersection de deux

cercles mais tu ne sais pas comment faire. . . Pas de panique, il existe une

solution. Je vais présenter celle disponible sous PSTricks, même s’il en existe

une aussi sous TikZ (comme elles portent le même nom, la documentation

est facile à trouver).

Il faut réaliser un clip. Le fonctionnement est très simple : tu définis la

zone d’intersection puis tu places un objet assez grand (comme un rectangle)

et paf ! Tu obtiens des Chocapics. . . bon ok, quand même pas mais le résultat

escompté est là et c’est le plus important.

Réaliser des intersections

% Compiler avec le moteur LaTeX

\documentclass[a4paper, 12pt]{report}

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage[dvipsnames]{xcolor}

\usepackage{pstricks}

\begin{document}

\begin{pspicture}[showgrid = true](5,5)

% Chemin de coupure / clip

\psclip{

\pscircle[linestyle = none](1,2){2}

\pscircle[linestyle = none](4,2){2}

}

211

Chapitre 16. Dessiner avec PSTricks

;A<

% linestyle = none pour ne pas le dessiner

% Remplir l'intérieur du chemin

\psframe\*[linecolor = Cyan](0,0)(4,4)

\endpsclip{}

% Affichage du contour du chemin de coupure

\pscircle(1,2){2}

\pscircle(4,2){2}

\end{pspicture}

\vspace{2\baselineskip}

\begin{pspicture}[showgrid = true](5,5)

% Chemin de coupure / clip

\psclip{

\rput{-30}(0,2){\psframe[linestyle = none](0,0)(3,2)}

% psrotate inopérant --> travailler avec rput (partir de

(0,0), rotation puis décalage)

\pscircle[linestyle = none](4,2){2}

}

% Remplir l'intérieur du chemin

\psframe\*[linecolor = Cyan](0,0)(4,4)

\endpsclip{}

\end{pspicture}

\end{document}

16.7 Extraction du contour d’une image

Il existe une image, une icône bien spécifique que tu as envie de réexploiter

mais elle est trop petite et les pixels sont apparents au moindre zoom. Ou bien

tu tiens à la créer toi-même sous LATEX pour obtenir une image vectorielle. . .

mais autant passer par un logiciel spécialisé parfois (Photoshop, InDesign,

etc.).

Même si le résultat n’est pas encore parfait, il est possible de vectoriser

une image, plus particulièrement sur des images monochromes. Le plus dur

et le seul point qui nous importe est l’obtention du contour de l’image. Dès

que nous avons les coordonnées des points, \psline suffit, quitte à ajouter

des options pour le remplissage.

password

######################################

###############**FLAG**##################

######################################

**CTF{J3FEC92!T=71O<FEA;"P@;F]W(&=O(')E<V]L=F4@=&AE(')E86P@0U1&}**

######################################

################**CTF**##################

######################################

212

Chapitre 16. Dessiner avec PSTricks

;A<

Par contre, pour réussir à obtenir les coordonnées du contour en question,

il faut bidouiller de la manière suivante :

❖ vectoriser l’image sous Inkscape, la lisser si nécessaire (enlever les

bosses superflues. . .) ;

❖ exporter le résultat au format .tex (option Enregistrer sous) ;

❖ ouvrir le code obtenu, vérifier les dimensions utilisées puis épurer le

code, c’est-à-dire ne garder que les coordonnées et enlever les commandes s’il y en a ;

❖ simplifier les coordonnées (beaucoup de décimales pas forcément utiles,

surtout à la vue des dimensions utilisées). Possibilité de laisser ce traitement à un algorithme (proposition d’un algorithme Python ci-après),

❖ insérer les coordonnées obtenues dans le code LATEX final de ton image ;

❖ bien reporter les dimensions ou les ajuster si besoin. Par exemple :

\psset{xunit = 0.5pt, yunit = 0.5pt}

Normalement, le résultat n’est pas trop moche. Tu peux le lisser sous

PSTricks sans effort en utilisant la commande \pscurve au lieu de \psline.

À toi de jouer ! 2

Simplifier les coordonnées : un algorithme Python

1 def reduction(chaine, nombre) :

2 """Réduire la valeur des décimales à "nombre" d'une

liste de coordonnées (x,y) (variable chaine)"""

3 resultat = ""

4 i = 0

5 while i <= len(chaine) - 1 :

6 # Coordonnée x

7 while chaine[i] != "." :

8 # On implémente et on cherche le point (séparateur

des décimales)

9 resultat = resultat + chaine[i]

2. Yu-Gi-Oh oh oh. . .

213

Chapitre 16. Dessiner avec PSTricks

;A<

10 i = i + 1

11 for j in range(0, nombre + 1) :

12 # On implémente la quantité de décimales voulues (

nombre)

13 resultat = resultat + chaine[i + j]

14 i = i + nombre + 1

15 while chaine[i] != "," :

16 # On a implémenté le nombre souhaité de décimales

17 # --> aller à l'autre coordonnées

18 i = i + 1

19

20 # Coordonnée y

21 while chaine[i] != "." : # Idem

22 resultat = resultat + chaine[i]

23 i = i + 1

24 for j in range(0, nombre + 1) : # Idem

25 resultat = resultat + chaine[i + j]

26 i = i + nombre + 1

27 while chaine[i] != ")" : # Idem

28 i = i + 1

29 resultat = resultat + chaine[i]

30 i = i + 1

31 print(resultat)

32

33 chaine = "(13.10458,13.112)(13.10458,13.112)

(13.10458,13.112)"

34 nombre = 2

35 reduction(chaine, nombre)

214

Chapitre 17

Dessiner avec TikZ

P

our faire des dessins, graphes, schémas, etc. avec LATEX sans avoir aucune

contrainte quant au moteur de compilation, c’est d’utiliser TikZ.

Si, comme moi, tu étais un habitué de PSTricks, il peut sembler déroutant

de passer à TikZ mais, avec la pratique, il devient facile de réaliser simplement

quelques figures. Mais ce n’est pas tout : TikZ est un bon compromis à

PSTricks.

Si la prise en main peut paraître compliquée de prime abord, son utilisation finit par devenir intuitive très rapidement. Ce package offre énormément

de possibilités, comme tu vas pouvoir le découvrir.

Et si jamais tu t’intéresses à la documentation officielle 1

, sache qu’il faut

mieux aller d’abord regarder le sommaire ou l’index. Avec plus de 1 000 pages

d’aide et de code, elle est plutôt bien fournie !

17.1 Démarrer sous TikZ

Règles de base

Tu vas difficilement pouvoir utiliser TikZ si tu ne charges pas le package associé : tikz. Comme nous le verrons plus tard, si tu dois charger

des fonctionnalités supplémentaires de TikZ, il faut utiliser la commande

\usetikzlibrary{<nom-bibliotheque>}, de préférence juste après avoir

chargé le package tikz.

Quant au dessin en lui-même, tout comme pour PSTricks, il faut charger

un environnement spécifique. Ici, il se nomme tikzpicture, et n’a pas besoin

d’options supplémentaires (comme la taille du cadre sous PSTricks). En effet,

1. Disponible sur le site du CTAN, directement sur : http://www.ctan.org/pkg/pgf.

215

Chapitre 17. Dessiner avec TikZ

;A<

TikZ produit toujours le résultat le plus compact possible, comme nous le

verrons dans un exemple juste après.

La règle capitale

Il existe une règle capitale sous TikZ : chaque commande

propre à TikZ se termine par un point-virgule “;”. Toujours. C’est

le seul point important à retenir, sous peine de ne pas comprendre

pourquoi ton code ne fonctionne pas.

Sous TikZ, s’il est plus courant de travailler avec des coordonnées cartésiennes (x, y), sache aussi que les coordonnées polaire (θ : R) sont disponibles,

écrites dans le même format que précédemment.

Il est aussi possible de définir des points, avec la commande \coordinate.

Sa syntaxe est la suivante, même si nous aurons l’occasion de revenir sur cette

commande par la suite :

\coordinate (<nom>) at (<coord>);

Un premier dessin

Pour commencer en douceur, le tracé d’un trait sous TikZ se fait de la

manière suivante :

\draw (x0,y0) -- (x1,y1);

La commande \draw annonce un tracé. Les points à relier par un trait sont

donc séparés par un double tiret “--”. Notons aussi au passage l’utilisation

du point-virgule “;” en fin de ligne, comme annoncé.

Il existe des fonctions propres à TikZ pour tracer un rectangle ou un

cercle. Il faut continuer d’utiliser la commande \draw au préalable :

\draw (x0,y0) rectangle (x1,y1);

\draw (x,y) circle (R);

Il est aussi possible d’augmenter l’épaisseur du trait ou de changer sa

couleur grâce à des options à introduire entre crochets “[]”, de la manière

216

Chapitre 17. Dessiner avec TikZ

;A<

suivante :

\draw[<options>] ...;

Je ne vais pas commencer à lister toutes les options possibles et envisageables. Les plus basiques sont présentées ci-après. Les autres sont à chercher

en fonction des besoins. Bien, voici un premier exemple pour avoir un aperçu

concret des bases :

Démarrer sous TikZ

% Ajout au PREAMBULE

%\usepackage{tikz}

TikZ produit toujours le résultat le plus compact possible :

\\

% Rendu final identique entre les 2 codes

\hspace\*{\fill}

\begin{tikzpicture}

\draw (0,0) -- (1,1); % Trait entre (0,0) et (1,1)

\end{tikzpicture}

\hfill

\begin{tikzpicture}

\draw (2,2) -- (3,3); % Trait entre (2,2) et (3,3)

\end{tikzpicture}

\hspace\*{\fill}

TikZ produit toujours le résultat le plus compact possible :

Des formes simples

\begin{tikzpicture}

\draw (0,0) rectangle (1,1);

217

Chapitre 17. Dessiner avec TikZ

;A<

\end{tikzpicture}

\hfill

\begin{tikzpicture}

\draw (0,0) rectangle (2,3);

\end{tikzpicture}

\hfill

\begin{tikzpicture}

\draw (1,1) circle (0.5);

\end{tikzpicture}

\hfill

\begin{tikzpicture}

\draw circle (1.3);

% Si coordonnées vides, (0,0) par défaut

\end{tikzpicture}

!

Une question ?

« Entre tes rectangles et tes cercles, l’origine (0, 0) n’est jamais

au même endroit. Quelle est cette diablerie ? »

En effet, je comprends ton questionnement. J’ai pris le parti pour

ces exemples d’utiliser des environnements tikzpicture distincts

pour chaque forme et pour pouvoir les espacer.

Comme TikZ produit le résultat le plus compact et l’affiche sur

la même ligne de base (environnement “alignés” ici), les origines sont

décalées. Tu peux constater que tout est en ordre si je réunis toutes

les commandes dans un même dessin :

218

Chapitre 17. Dessiner avec TikZ

;A<

!

Utilisation d’un seul environnement

\begin{tikzpicture}

\draw (0,0) rectangle (1,1);

\draw (0,0) rectangle (2,3);

\draw (1,1) circle (0.5);

\draw circle (1.3);

\end{tikzpicture}

Reprenons avec d’autres exemples minimalistes, pour te montrer les options de base de TikZ :

Un peu de couleur

\hspace\*{\fill}

\begin{tikzpicture}

% Forme courte et implicite

\draw[blue] (0,0) -- (1,1);

% Forme complète (nom option)

\draw[color = orange] (2,1) -- (3,0);

\end{tikzpicture}

\hfill

\begin{tikzpicture}

% Idem pour un contour (fermé)

\draw[red] (0,0) rectangle (1,1);

\draw[color = green] (2,1) rectangle (3,0);

% Autre possibilité (nuancer avec le remplissage)

\draw[draw = purple] (4,0) rectangle (5,1);

\end{tikzpicture}

\hspace\*{\fill}

219

Chapitre 17. Dessiner avec TikZ

;A<

Un peu de remplissage

% Ligne 1

\begin{center}

\begin{tikzpicture}

% Remplissage rouge avec contour noir (par défaut)

\draw[fill = red] (0,0) rectangle (1,1);

% Remplissage pur (sans contour)

\fill[color = green] (2,1) rectangle (3,0);

\fill[fill = purple] (4,0) rectangle (5,1);

\end{tikzpicture}

\end{center}

% Ligne 2

\begin{center}

\begin{tikzpicture}

\draw[red, fill = blue] (0.5,0.5) circle (0.5);

\draw[draw = green, fill = orange] (2.5,0.5) circle (0.5);

% Nouvelle commande : contour et remplissage

\filldraw (4.5,0.5) circle (0.5);

\filldraw[brown] (6.5,0.5) circle (0.5);

% Personnalisation toujours possible

\filldraw[pink, draw = gray] (8.5,0.5) circle (0.5);

\end{tikzpicture}

\end{center}

Nous pouvons constater que si une couleur seule est renseignée, TikZ

l’associe automatiquement à l’option draw. Il existe même un choix supplémentaire pour indiquer qu’il ne faut pas mettre de couleur 2

: draw = none

ou fill = none.

2. Le fond n’est pas toujours blanc donc choisir white n’est pas toujours judicieux.

220

Chapitre 17. Dessiner avec TikZ

;A<

Changement d’épaisseur et de trait

\begin{center}

\begin{tikzpicture}

% Epaisseur du trait : line width = <longueur>

\draw[line width = 1.3mm] (0,0) -- (1,1);

\draw[line width = 5pt] (2,1) -- (3,0);

% Unité par défaut (options) : le point "pt"

\draw[line width = 8] (4,0) -- (5,1);

% N.B. --> unité par défaut (coordonnées) : "cm"

\end{tikzpicture}

\end{center}

\begin{center}

\begin{tikzpicture}

% Tailles prédéfinies

\draw[thin] (0,0) -- (1,1);

% Nouveau trait : tiret

\draw[thick, dashed] (2,1) -- (3,0);

% Nouveau trait : en pointillé

\draw[ultra thick, dotted] (4,0) -- (5,1);

\end{tikzpicture}

\end{center}

!

Renseigner les options dans le bon ordre

TikZ lit les options indiquées de gauche à droite et les applique

une par une, dans cette ordre de lecture. Sur le PDF, le remplissage

des options est donc à faire en LIFO : Last In First Out soit « dernier

arrivé premier servi » !

Tu peux t’en rendre compte très facilement avec le cas suivant :

221

Chapitre 17. Dessiner avec TikZ

;A<

!

Exemple

\begin{center}

\begin{tikzpicture}

% Cas 1

\filldraw[pink, draw = gray, line width = 3pt] (0,0)

circle (0.5);

% Cas 2 <> Cas 1

\filldraw[draw = gray, line width = 3pt, pink] (2,0)

circle (0.5);

\end{tikzpicture}

\end{center}

Tu as tout compris ? Il existe plein d’options extrêmement pratiques mais

la couleur et l’épaisseur du trait sont généralement celles les plus couramment

utilisées au début. Il existe aussi des épaisseurs prédéfinies, qui fonctionnent

très bien et évitent de perdre du temps à trouver la “bonne” épaisseur :

→ ultra thin : 0.1pt ;

→ very thin : 0.2pt ;

→ thin : 0.4pt (défaut) ;

→ semithick : 0.6pt ;

→ thick : 0.8pt ;

→ very thick : 1.2pt ;

→ ultra thick : 1.6pt.

Essayons maintenant de tracer des figures un peu plus complexes désormais, avec des coordonnées polaires pour changer un peu et les manipuler.

17.2 Un polygone régulier

Je pense que tu dois avoir déjà entendu parler d’un polygone régulier.

Pour faire simple et éviter de faire mon pédant trop longtemps, il s’agit

d’une figure géométrique fermée, à N côtés de même longueur.

Une façon très simple d’en créer consiste à passer par des coordonnées

polaires. En effet, les sommets Si d’un polygone régulier sont tous placés sur

222

Chapitre 17. Dessiner avec TikZ

;A<

un cercle de centre O quelconque, de rayon R et la droite (OSi) forme un

angle de θi =

i × 360

N

avec l’axe des abscisses.

Les coordonnées polaires du sommet Si s’écrivent donc naturellement

(θi

: R).

Pour revenir sur l’utilisation de TikZ, il faut savoir que toute figure définie

par des traits et dont le point d’arrivée coïncide avec le point de départ (figure

fermée) doit se conclure de la manière suivante : -- cycle;.

Cette commande permet de proprement fermer la figure. Je te laisse aller

faire des recherches ou des essais pour voir la différence avec une fermeture

manuelle. Sinon, un petit exemple fait main, parce que cela me fait plaisir :

(a) Ce qu’il faut faire (fermeture avec

cycle)

(b) Ce qu’il ne faut pas faire (fermeture manuelle)

Bien, arrêtons-nous là concernant les détails et considérations techniques.

Allons plutôt dessiner un polygone régulier, comme un triangle équilatéral,

pour commencer simplement :

Un triangle équilatéral

% Triangle équilatéral, inscrit dans un cercle de rayon R

% Coordonnées polaires ==> centre (0,0)

\begin{center}

\begin{tikzpicture}

% Rayon R choisi arbitrairement à 2cm

\draw (90:2) -- (210:2) -- (330:2) -- cycle;

% Une autre possibilité

\draw[green] (60:2) -- (180:2) -- (300:2) -- cycle;

\draw[red] circle (2); % Pas de centre ==> (0,0) par défaut

\end{tikzpicture}

\end{center}

223

Chapitre 17. Dessiner avec TikZ

;A<

Avouons que, dans ce cas de figure, l’utilisation des cordonnées polaire est

plus pratique que de devoir placer 2 points et calculer la position du dernier,

surtout si les calculs ne donnent pas une valeur exacte. Ici, notre triangle est

bel et bien équilatéral.

Le tracé peut aussi s’envisager avec des points définis à l’avance :

Définir des points

\begin{center}

\begin{tikzpicture}

\coordinate (A) at (30:2);

\coordinate (B) at (150:2);

\coordinate (C) at (270:2);

\draw (A) -- (B) -- (C) -- cycle;

\end{tikzpicture}

\end{center}

224

Chapitre 17. Dessiner avec TikZ

;A<

17.3 Automatiser les dessins

Coordonnées absolues et relatives

Bon, tracer un triangle équilatéral, c’est bien. Tracer un hexagone, avec

un copier-coller et un peu de patience, c’est faisable. Un tridécagone (polygone régulier à 13 côtés). . . bon, rien d’impossible mais le copier-coller et les

modifications ne constituent clairement pas une solution optimale !

Fort heureusement, il existe le principe des coordonnées absolues et relatives.

Pour faire simple, tracer un dessin grâce à une série de coordonnées absolues revient à connaître les positions de toutes les coordonnées par rapport

à un repère, l’origine (0, 0) généralement mais il peut aussi s’agir d’un autre

point.

Avec les coordonnées relatives, peu importe la position exacte de tous les

points : il suffit juste de connaître la position d’un point par rapport à un

autre !

Sous TikZ, les coordonnées absolues ne requiert aucune option spécifique,

hormis la position du point. Les coordonnées relatives sont facilement reconnaissables grâce au “++” et il existe un mix des deux, un peu subtil, qui utilise

un “+”. L’aide officielle est assez explicite à ce sujet :

«

You can add a single + sign in front of a coordinate or two of

them as in +(1cm,0cm) or ++(0cm,2cm). Such coordinates are

interpreted differently.

The first form means “1cm upwards from the previous specified

position” ; the second means “2cm to the right of the previous

specified position, making this the new specified position.”

»

Concrètement, le tracé (A) --+ (xB,yB) --+ (xC,yC) définit les points

B et C par rapport au point A.

Par contre, le tracé (A) --++ (xB,yB) --++ (xC,yC) définit le point B

par rapport au point A puis le point C par rapport au point B (nouvelle origine

pour le déplacement suivant).

Bien, je pense qu’un petit exemple ne sera pas de trop pour aborder cette

notion :

225

Chapitre 17. Dessiner avec TikZ

;A<

Coordonnées absolues et relatives

% Je reviendrai sur les "node" par la suite

% Ici, ils permettent d'avoir un point de repère pour

distinguer le départ du tracé (en rouge) de sa fin (en

bleu)

% Sans les + ou ++

\begin{tikzpicture}

\draw[gray, dotted] (0,-1) grid (3,1); % Une trame de fond,

pour aider

\draw (0,0) node[circle, fill = red, inner sep = 2pt] {} --

(1,1) -- (2,0) -- (0,-1) node[circle, fill = blue, inner

sep = 2pt] {}; % Le point de départ est toujours le point

à partir duquel est appliqué le déplacement

\end{tikzpicture}

\hfill

% Avec le +

\begin{tikzpicture}

\draw[gray, dotted] (0,-1) grid (3,1); % Une trame de fond,

pour aider

\draw (0,0) node[circle, fill = red, inner sep = 2pt] {} --

(1,1) --+ (2,0) --+ (0,-1) node[circle, fill = blue, inner

sep = 2pt] {}; % Le dernier point sans "+" -- (1,1) ici

-- est toujours le point à partir duquel est appliqué le d

éplacement

\end{tikzpicture}

\hfill

% Avec le ++

\begin{tikzpicture}

\draw[gray, dotted] (0,-1) grid (3,1); % Une trame de fond,

pour aider

\draw (0,0) node[circle, fill = red, inner sep = 2pt] {} --++

(1,1) --++ (2,0) --++ (0,-1) node[circle, fill = blue,

inner sep = 2pt] {}; % Chaque nouveau point est le point

de départ pour le déplacement d'après

\end{tikzpicture}

226

Chapitre 17. Dessiner avec TikZ

;A<

Et cette méthode s’applique aussi pour les coordonnées polaires ! Appliquons alors cette découverte pour nos polygones réguliers. Après tout, il s’agit

de prendre le point précédent et de le faire pivoter du bon angle :

Coordonnées relatives polaires

% Cas d'un triangle équilatéral

\begin{tikzpicture}

\draw (0,0) -- (2,0) --++ (120:2) -- cycle;

\end{tikzpicture}

\hfill

% Cas d'un carré

\begin{tikzpicture}

\draw (0,0) -- (2,0) --++ (90:2) --++ (180:2) -- cycle;

\end{tikzpicture}

\hfill

% Cas d'un pentagone

\begin{tikzpicture}

\draw (0,0) -- (2,0) --++ (72:2) --++ (144:2) --++ (216:2) --

cycle;

\end{tikzpicture}

Un exemple plus concret parlera peut-être à certains :

227

Chapitre 17. Dessiner avec TikZ

;A<

Un peu de physique

% Ou comment simplifier le tracé d'un amortisseur

\begin{center}

\begin{tikzpicture}

\coordinate (O) at (0,0); % Possibilité de changer le 0,0 en

argument d'une nouvelle commande...

\draw (O) --++ (2,0) --++ (0,-1) --++ (2,0) ++ (-2,1) --++

(0,1) --++ (2,0) node[above left] {\Large{}$\mu$} ++

(-1,0) --++ (0,-2) ++ (0,1) --++ (2,0);

% Utilisation de "++" sans "--" pour déplacer la coordonnée

relative (on rebrousse chemin dans le tracé) sans tracer

un trait

\end{tikzpicture}

\end{center}

µ

!

Une question ?

« Ton astuce est amusante pour tracer le polygone en polaire ou l’amortisseur mais ce n’est toujours pas pratique. Il faut

quand même changer à la main les valeurs pour chaque polynôme. . . »

En effet. . . mais j’allais justement annoncer une magnifique solution automatisée !

Variables et boucle for

Il existe trois outils que j’ai découverts suite à mon passage à TikZ et qui

se révèlent très utiles pour automatiser le tracé de dessins :

→ la définition de variable : tu peux créer toi-même ta propre variable

sous LATEX

3

. Appliquée à TikZ, tu peux l’associer en tant que nombre

3. Très exactement, il s’agit d’une macro. J’apporterai sûrement un correctif et une

228

Chapitre 17. Dessiner avec TikZ

;A<

(nombre de côtés d’un polygone régulier par exemple) ou en tant que

longueur (rayon du cercle dans lequel le dit polygone est inscrit).

Il suffit d’utiliser la commande suivante :

\def\nom{valeur}

→ le calcul de nouvelles variables : propre à TikZ, cette possibilité

peut parfois servir.

Par exemple, nous souhaitons dessiner un polygone régulier inscrit dans

un cercle de rayon fixé, sans connaître la valeur d’un côté (même si c’est

bien plus simple de considérer la taille d’un cercle pour l’affichage).

Si tu désires avoir un polygone avec une taille d’arête bien spécifique, il

faut calculer le rayon ! Pour ce faire, il faut alors utiliser la commande :

\pgfmathsetmacro\nom{<calcul>}

Il est aussi possible d’utiliser des variables déjà définies pour les intégrer dans le calcul. Les possibilités offertes deviennent alors très intéressantes ;

→ la boucle for : comme avec un langage de programmation, il est

possible d’indiquer à LATEX, et plus particulièrement à TikZ dans notre

cas, des tâches répétitives. La formulation est la suivante :

\foreach \<var> in {1,...,N} {<boucle>}

J’ai mis {1,...,N} pour l’exemple générique mais tu peux mettre n’importe quelle valeur numérique, comme {2,3,4}, ou même des lettres !

C’est des fois pratique pour jongler avec des coordonnées.

En guise d’exemple, voici une solution simple qui fonctionne. Il y a sûrement encore moyen de l’améliorer, comme permettre à chaque trait d’avoir

une couleur différente (avec cycle en fin de ligne sinon c’est moche) mais

elle fonctionne déjà plutôt bien :

explication plus poussée lors de la prochaine mise à jour de ce guide et après quelques

recherches.

229

Chapitre 17. Dessiner avec TikZ

;A<

Une solution automatisée

% Un polygone régulier

\begin{center}

\begin{tikzpicture}

% Paramétrage

\def\poly{13} % Nombre entier supérieur à 1

% Limite de calcul LaTeX fixée à 16 383...

\pgfmathtruncatemacro\polyg{\poly - 1}

\def\R{2} % 0.13\linewidth est aussi une distance

% Tracé du polygone

\draw[orange] (90:\R) \foreach \i in {1,...,\polyg} {-- (90-\i

/\poly\*360:\R)} -- cycle; % Usage de \polyg pour pouvoir

bien fermer avec "cycle"

\end{tikzpicture}

\end{center}

La gestion des unités

Il peut arriver que tu définisses une variable mais que sa valeur

ne donne pas le résultat attendu, en terme de taille. Par exemple, un

rayon \def\R{50} de 50pt ou 50mm au lieu de 50cm par défaut, un

peu grand, surtout sur une feuille A4 ; ou encore, une épaisseur de trait

\def\sep{13} de 13mm au lieu de 13pt par défaut.

Seulement, écrire circle (\R{}pt) ou line width = \sep mm

ne fonctionne pas toujours car LATEX n’arrive pas à combiner une

variable avec du texte. . .

Heureusement, il existe donc un moyen très simple de résoudre ce

problème. Il faut définir une variable unité : \def\unit{<unité>}. Par

230

Chapitre 17. Dessiner avec TikZ

;A<

exemple, nous pouvons écrire \def\unit{pt} ou \def\mm{mm} s’il y a

plusieurs unités et que tu ne veux pas les confondre.

Il faut ensuite écrire, par exemple, circle (\R\unit) ou

line width = \sep\mm, et le tour est joué !

!

Limite de calcul sous TikZ

Avec les commandes \def et \pgfmathsetmacro, il existe une limite de calcul, fixée à 16 383, très exactement 2

30 − 1

2

16 . Du coup, si

tu veux tracer un polygone de 17 000 côtés, c’est impossible. Et je

n’aborde pas l’intérêt d’un tel tracé : autant utiliser un cercle dans

ce cas !

Généralement, pour des cas raisonnables, il ne devrait pas y avoir

de problème mais il est bon de connaître cette notion.

Dans le cas où une telle erreur apparaît, le compilateur devrait

afficher l’erreur « ! Dimension too large. ». Mais il peut aussi

arriver que cette limite apparaisse alors que les calculs ne

dépassent pas la valeur interdite !

Par exemple, trace un polygone de 50 côtés avec mon code précédent et essaye les deux possibilités suivantes dans la boucle for :

→ {-- (90-\i/\poly\*360:\R)} : aucun problème,

→ {-- (90-\i\*360/\poly:\R)} : problème . . . alors que, d’un

point de vue purement formel, le calcul est le même !

De ce que j’ai compris, il s’agit d’une erreur due à un dépassement

de pile (stack overflow) sous TikZ. Pour l’éviter, il faut toujours

privilégier les divisions au début du calcul.

17.4 Dessiner des figures mathématiques

Je ne vais pas m’attarder sur cette section, juste donner deux pistes de

recherche. Si tu as beaucoup de figures géométriques à dessiner, et surtout des

figures mathématiques, avec beaucoup de sommets, des intersections, etc., tu

peux :

→ utiliser le logiciel gratuit GeoGebra (https://www.geogebra.org/) et

exporter les figures en code TikZ ;

231

Chapitre 17. Dessiner avec TikZ

;A<

→ utiliser le package tkz-euclide, qui possède une documentation bien

fournie et beaucoup de commandes intéressantes.

Et voici un premier exemple de mise en bouche pour tracer une fonction :

Une fonction bien connue et son inverse

\begin{tikzpicture}[samples = 130]

\draw[->] (-2,0) -- (2,0) node[right] {$x$};

\draw[->] (0,-2) -- (0,2) node[above, text = cyan] {$\sin x$};

\draw (-1.57,1mm) -- (-1.57,-1mm) node[below] {\footnotesize

$-\frac{\pi}{2}\phantom{-}$};

\draw (1.57,1mm) -- (1.57,-1mm) node[below] {\footnotesize $

\frac{\pi}{2}$};

\draw (1mm,-1) -- (-1mm,-1) node[left] {$-1\strut$};

\draw (1mm,1) -- (-1mm,1) node[left] {$1\strut$};

\node[below right] at (0,0) {$0$};

\draw[thick, color = cyan, domain = -1.57:1.57] plot ({\x},{

sin(deg(\x))}); % deg pour conversion

\end{tikzpicture}

\hfill

\begin{tikzpicture}[samples = 130]

\draw[->] (-2,0) -- (2,0) node[right] {$x$};

\draw[->] (0,-2) -- (0,2) node[above, text = red] {$\arcsin x

$};

\draw (-1,1mm) -- (-1,-1mm) node[below] {$-1\phantom{-}$};

\draw (1,1mm) -- (1,-1mm) node[below] {$1$};

\draw (1mm,-1.57) -- (-1mm,-1.57) node[left] {\footnotesize $-

\frac{\pi}{2}\strut$};

\draw (1mm,1.57) -- (-1mm,1.57) node[left] {\footnotesize $

\frac{\pi}{2}\strut$};

\node[below right] at (0,0) {$0$};

\draw[thick, dashed, color = red, domain = -1:1] plot ({\x},{

rad(asin(\x))}); % rad pour conversion

232

Chapitre 17. Dessiner avec TikZ

;A<

\end{tikzpicture}

x

sin x

−

π

2

π

2 −1

1

0

x

arcsin x

−1 1

−

π

2

π

2

0

Comme tu peux le constater, je trace le repère et les points remarquables

(traits sur le repère) à la main. Puis, j’utilise plot ({\x},{f(\x)}) pour le

tracé de la fonction.

Il existe sûrement d’autres syntaxes pour tracer des courbes polaires ou

avec plusieurs variables. Je te laisse te documenter à ce sujet si tu en as

besoin. Le package pgfplots peut proposer des éléments de solution.

Bien, maintenant que nous connaissons le fonctionnement de TikZ et

l’avons un peu manipulé, voyons maintenant des méthodes élégantes pour

gérer facilement la forme de tes dessins, et donc toutes les options disponibles.

17.5 Gestion des styles

!

Chargement du package xcolor

Avant de commencer à parler de style, je dois aborder un cas

assez délicat : l’importation du package xcolor, et encore plus de ses

options, dont dvipsnames pour ma part. Qui dit style pense alors à

couleur : xcolor est alors indispensable. Mais il peut se produire une

erreur à la compilation.

Il faut déjà savoir qu’il faut toujours charger xcolor avant

tikz. Mais, dans certains cas (utilisation d’autres packages principalement), il peut arriver qu’il y ait un conflit et que l’erreur

Option clash for package xcolor surgisse.

233

Chapitre 17. Dessiner avec TikZ

;A<

!

Il existe alors 2 solutions :

→ identifier le package qui pose problème et charger xcolor

avant. Dans mon cas, je me suis rendu compte que le package

tcolorbox était la source des erreurs : j’appelle donc xcolor

avant ;

→ appeler la commande suivante avant l’appel de la classe

documentclass :

\PassOptionsToPackage{dvipsnames}{xcolor}

Maintenant que tu es au courant de cette astuce, je ferme la parenthèse et je passe aux styles.

Imaginons un instant que nous avons plein de traits, de rectangles et de

cercles à tracer. Bref, pleins d’éléments qui requiert d’utiliser beaucoup de

\draw. Nous voulons aussi que tous ces éléments aient le même format (couleur, épaisseur de trait, etc.), pour homogénéiser le rendu.

Il est possible de définir un style global pour un dessin, lors de l’appel de

l’environnement tikzpicture. Au lieu d’écrire \draw[<options>] à chaque

fois et de devoir tout changer manuellement, il est possible d’ajouter des

options à l’environnement de la manière suivante :

\begin{tikzpicture}[<options>]

Si nous avons besoin de définir plusieurs styles distincts, c’est possible de

regrouper toutes les options dans un nom de style pour pouvoir les appeler

directement. Il faut alors utiliser la syntaxe suivante :

<nom-style>/.style = {<options>}

Tu peux déclarer ton style soit lors de l’appel de l’environnement tikzpicture,

soit avec la commande \tikzset{<def-style>}, avant d’appeler le style en

question dans les options : \draw[<nom-style>].

Enfin, il est toujours possible de procéder à des changements ponctuels

dans les options d’un \draw. Placés après un style, ils prédomineront à

coup sûr (lecture de gauche à droite des options et la dernière option lue est

appliquée).

D’une certaine manière, nous pouvons voir <nom-style> comme une variable qui contient du texte que LATEX et TikZ se chargent de copier-coller à

234

Chapitre 17. Dessiner avec TikZ

;A<

chaque appel du style.

Un petit exemple pour bien comprendre, comme d’habitude :

Les styles sous TikZ

% Paramètre défini globalement

\def\R{1.5}

% Style global

\begin{tikzpicture}[thick, red, dashed]

\draw circle (\R);

\draw (\R,0) --++ (-2\*\R,0);

\draw (0,\R) --++ (0,-2\*\R);

\end{tikzpicture}

\hfill

% Styles locaux

\begin{tikzpicture}[thick, red, dashed]

\draw[solid, thin] circle (\R);

% solid = trait plein

\draw (\R,0) --++ (-2\*\R,0);

% Bien mettre un \* pour le calcul

% Ne pas confondre avec les longueurs : 0.5\linewidth licite

\draw[cyan, ultra thick] (0,\R) --++ (0,-2\*\R);

\end{tikzpicture}

\hfill

% Style groupé

\begin{tikzpicture}[thick, red, dashed]

\tikzset{pointille/.style = {purple, line width = 4pt, line

cap = round, dash pattern = on 0pt off 2.5\pgflinewidth}}

% Style dotted pas très "dot" --> utilisation de line cap &

dash pattern

\draw circle (\R);

\draw[pointille] (\R,0) --++ (-2\*\R,0);

% Changement d'un style déjà défini

\draw[pointille, orange] (0,\R) --++ (0,-2\*\R);

\end{tikzpicture}

235

Chapitre 17. Dessiner avec TikZ

;A<

Bien évidemment, ici, le code est très simple et cette notion devient intéressante quand tu as beaucoup de \draw, ou quand tu te rends compte

que tu fais beaucoup de changements dans les options. Il devient alors plus

intéressant de les automatiser avec des styles.

!

Appel de \tikzset

Dans l’exemple fourni, le style pointille est défini à l’intérieur

de l’environnement tikzpicture. Par conséquent, il n’est utilisable

que pour cet environnement. En l’état, impossible de l’appeler dans

un autre environnement tikzpicture.

Toutefois, \tikzset est utilisable à n’importe quel endroit de ton

code LATEX. Tu peux donc l’appeler en-dehors de l’environnement

tikzpicture. Il sera alors disponible pour toutes les figures à venir.

Tu peux donc même définir tes styles dans le préambule. C’est

plus facile à gérer (regroupement de toutes les commandes dans un

même endroit au sein de ton code) et tu peux t’en servir à volonté

par la suite !

Maintenant que la gestion de la mise en forme avec les styles est bien

définie et que nous savons tracer quelques figures élémentaires, pimentons un

peu les possibilités. Ajoutons du texte !

17.6 Insérer du texte

Il n’y a qu’une seule façon d’écrire dans un dessin réalisé sous TikZ :

utiliser les node. Très exactement, les node permettent de placer à peu près

tout et n’importe quoi à l’endroit souhaité dans le dessin, en particulier du

texte.

236

Chapitre 17. Dessiner avec TikZ

;A<

Un node s’appelle par une commande, selon la syntaxe suivante :

\node[<options>] (<nom>) at (<coord>) {<texte>};

Il existe de multiples possibilités en ce qui concerne les options d’un node

(<options>). Celles que j’utilise fréquemment sont les suivantes et sont rappelées en Table A.8 (annexes p. 269) :

→ circle (rectangle par défaut) : pour avoir un cercle comme cadre au

lieu du rectangle. Différents formats sont disponibles et sont explicités

dans les exemples un peu après ;

→ draw = <color> (black par défaut) : pour afficher le cadre du node et

définir sa couleur ;

→ fill = <color> : la couleur de remplissage du cadre ;

→ text = <color> : la couleur du texte ;

→ font = <mise-en-forme> : pour mettre en forme le texte (\bfseries,

\itshape, \small, etc.) ;

→ align = <position>, avec <position> qui peut prendre les valeurs

left, center ou right : pour aligner horizontalement le texte à l’intérieur du node ;

→ inner sep = <taille> : espacement entre le texte et le bord du node ;

→ outer sep = <taille> : espacement entre le bord du node et les

autres éléments ;

→ text width = <taille> : largeur de la boîte (invisible) dans laquelle

est placée le texte. Si text width est inférieur à minimum width, la

boîte en question est centrée.

Toutefois, le texte à l’intérieur peut continuer à être excentré si text width

est supérieur à la taille minimale du texte. Dans ce cas, utiliser l’option

align pour centrer à ta convenance ;

→ minimum width = <taille> : largeur minimale du cadre ;

237

Chapitre 17. Dessiner avec TikZ

;A<

→ minimum height = <taille> : hauteur minimale du cadre ;

→ <position> (above, below, left ou right) : pour positionner le node

par rapport à <coord> ;

→ rotate = <angle> : pour faire pivoter le node.

Un premier exemple d’application peut prendre la forme suivante :

Utilisation des node

\begin{center}

\begin{tikzpicture}

% Cas simple

\draw (0,0) -- (1,0);

\node at (0.5,0.5) {Texte};

% Affichage de la bordure rectangulaire du node

\node[draw] at (0.5,2) {some text};

% Changement du cadre avec circle (rectangle par défaut)

\node[draw, circle, align = left] at (4,1.5) {Texte sur 2

lignes \\ Retour ligne manuel};

% Cas plus complet

\node[draw = Green, line width = 8pt, fill = red!30, font =

\scshape, text = gray!75!black, thick, minimum width = 4.5

cm, text width = 4cm, minimum height = 2cm, align = center

, rotate = 90] at (8,1.5) {Texte centré \\ sur 2 lignes};

\end{tikzpicture}

\end{center}

Texte

some text Texte sur 2 lignes

Retour ligne manuel

Texte centré

sur 2 lignes

238

Chapitre 17. Dessiner avec TikZ

;A<

!

Saut de ligne dans un node

Le saut de ligne manuel avec \\ est licite et fonctionnel dans un

node si et seulement si l’option align = <position> est utilisée.

Sans cette option, TikZ écrit le texte seul et n’interprète pas le

saut de ligne. Avec cette option, il doit placer ton texte dans une

parbox bien paramétrée avec le centrage indiqué, ce qui rend le saut

de ligne possible.

Mais c’est mon hypothèse personnelle : il faudrait aller lire la

documentation voire le code source pour s’en assurer.

Comme indiqué, les node peuvent prendre différentes formes, dont voici

une liste non exhaustive :

❖ formes simples : rectangle, circle, ellipse, diamond, circle split,

forbidden sign, cross out, strike out ;

❖ formes plus “complexes” (options supplémentaires) :

→ regular polygon & regular polygon sides = 5,

→ star, star points = 7 & star point ratio = 0.8.

Certaines formes sont disponibles grâce à différentes bibliothèques TikZ, à

charger après l’appel du package. Pour avoir un aperçu des différents formats

disponibles, c’est par ici avec un exemple minimal :

Les différents formats de node

% Ajout au PREAMBULE

%\usepackage{tikz}

%\usetikzlibrary{shapes.geometric, shapes.misc, shapes.

multipart, shapes.symbols}

\begin{tikzpicture}

% Affichage sous forme d'un tableau (syntaxe identique)

\matrix[nodes = {draw, ultra thick, fill = blue!20}, row sep =

3mm, column sep = 4mm] {%

\node[draw = none, fill = none] {Plain node}; & \node[

rectangle] {Rectangle}; & \node[circle] {Circle}; \\

\node[ellipse] {Ellipse}; & \node[circle split] {Circle

\nodepart{lower} split}; & \node[forbidden sign, text

239

Chapitre 17. Dessiner avec TikZ

;A<

width = 4em, text centered] {Forbidden sign}; \\

\node[diamond] {Diamond}; & \node[cross out] {Cross out};

& \node[strike out] {Strike out}; \\

};

\end{tikzpicture}

\begin{tikzpicture}[note/.style = {draw = none, fill = none,

right}]

\matrix[nodes = {draw, ultra thick, fill = blue!20}, row sep =

3mm, column sep = 4mm] {%

\node[regular polygon, regular polygon sides = 5] {$n =

5$}; & \node[regular polygon, regular polygon sides = 7]

{$n=7$}; & \node[regular polygon, regular polygon sides =

9] {$n=9$}; & \node[note]{Regular polygon}; \\

\node[star, star points = 4] {$p = 4$}; & \node[star, star

points = 7, star point ratio = 0.8] {$p=7$}; & \node[star

, star points = 9] {$p = 9$}; & \node[note]{Star}; \\

};

\end{tikzpicture}

Plain node Rectangle Circle

Ellipse Circle

split

Forbidden

sign

Diamond Cross out Strike out

240

Chapitre 17. Dessiner avec TikZ

;A<

n = 5 n = 7 n = 9 Regular polygon

p = 4 p = 7 p = 9 Star

Mais tu peux tout faire avec des node. Par exemple, tu peux les placer

à l’intérieur d’un chemin dessiné par un \draw pour ajouter de l’information

(texte ou symbole).

L’intérêt ? Pendant que tu traces ton dessin, tu associes l’information à

la coordonnée, au lieu d’ajouter le texte manuellement par la suite. C’est

très pratique si tu modifies ton dessin ou si les coordonnées sont difficiles à

déterminer.

En revanche, ton code est moins lisible à relire. . . Reprenons un ancien

exemple, qui devrait te sembler plus clair désormais :

Ajouter de l’information sur un tracé avec des node

\begin{center}

\begin{tikzpicture}[cercle/.style = {circle, inner sep = 2pt}]

\draw[gray, dotted] (0,-1) grid (3,1);

% Les node en cascade, c'est le nec plus ultra !

\draw (0,0) node[cercle, fill = red] {} node[left] {Début}

--++ (1,1) --++ (2,0) --++ (0,-1) node[cercle, fill = blue

] {} node[right] {Fin};

\draw (0,0) -- (1,-1) node[right, align = left] {Nouvelle \\

branche};

\end{tikzpicture}

\end{center}

241

Chapitre 17. Dessiner avec TikZ

;A<

Début Fin

Nouvelle

branche

Pour rappel, le saut de ligne manuel avec \\ est licite et fonctionnel dans

un node si et seulement si l’option align = <position> est utilisée.

Les options comme above ou below permettent d’ajuster la position du

node par rapport à la coordonnée à laquelle il se réfère. Des combinaisons

sont possibles, comme above left. Toutefois, il faut respecter un certain

ordre : left above ne fonctionne pas par exemple. 4

Bon, je crois avoir à peu près fait le tour en ce qui concerne la base

pour les node. Voyons une dernière application, plus poussée : la création de

graphes et de diagrammes.

17.7 Création de graphes et de diagrammes

Les points d’ancrage

Sous TikZ, un node est constitué de points d’ancrage, répartis de la manière suivante :

node N

× ×

×

× × ×

×

×

N.north N.north east

N.east

N.south west N.south N.south east

N.west

N.north west

×

× ×

×

N.13 (angle)

N.text N.base

N.center = N

Figure 17.2 – Vue d’un node et de ses points d’ancrage

4. Je n’ai pas d’astuce pour s’en souvenir. Je le sais et je fais attention à bien lire les

erreurs de compilation annoncées pour vite corriger le tir.

242

Chapitre 17. Dessiner avec TikZ

;A<

J’ai pris le cas traditionnel d’un node rectangulaire mais les points d’ancrage sont parfaitement définis pour toutes les autres formes, si besoin (cercle,

étoile, polygone régulier, etc.).

Voyons maintenant comment utiliser cette notion pour placer deux node

sous forme de boîte l’un par rapport à l’autre. Nous allons utiliser judicieusement les styles déjà évoqués et les points d’ancrage.

Cette méthode permet d’avoir à placer un seul node (référence) et tout

peut se faire relativement à ce dernier, ou relativement aux nouveaux node.

Une nouvelle option est alors adaptée à la situation : anchor = <ancre>.

De cette manière, tu peux spécifier le point d’ancrage sur lequel le node va

se fixer. Le point d’ancrage utilisé par défaut est center.

Style global aux node

Si nous avons déjà vu la syntaxe pour créer un style sous TikZ

et les configurations disponibles (style global ou local), il est possible

d’indiquer directement un style à tous les node, grâce à la syntaxe

suivante :

\begin{tikzpicture}[every node/.style = {<options>}]

% Dessiner !

\end{tikzpicture}

Mais cette option est à utiliser judicieusement : c’est pratique si

tous tes node sont des boîtes. C’est plus gênant à corriger si tu ajoutes

en milieu de chemin un node simple, pour écrire un commentaire par

exemple.

Après la théorie, un peu de pratique avec un premier code d’initiation :

Initiation aux points d’ancrage

\begin{center}

\begin{tikzpicture}[every node/.style = {draw = orange, very

thick, minimum width = 2cm, minimum height = 1cm}]

% Node "master" (référence)

\node (master) at (0,0) {Boîte maître};

243

Chapitre 17. Dessiner avec TikZ

;A<

% Node créé relativement à "master"

\node[anchor = west, xshift = 2cm] (boiteA) at (master.east) {

Boîte A};

% Positionnement de l'ancrage ouest de boiteA sur l'ancrage

est de "master"

% Décalage horizontal manuel avec xshift

\draw (master) -- (boiteA);

% TikZ trace le trait le plus simple pour relier 2 node

% Troisième node

% Décalage vertical manuel avec yshift

\node[draw = Green, fill = gray!50, minimum width = 2cm,

yshift = -2cm] (boiteB) at (boiteA.south west) {Boîte B};

% Tracé d'un trait "|-|" automatisé

% Création d'un node milieu

\path (master.south) -- (boiteB.north) coordinate [midway] (

middle);

\draw (master.south) |- (middle) -| (boiteB.north);

\end{tikzpicture}

\end{center}

Boîte maître Boîte A

Boîte B

Bon, d’accord, je triche un peu avec les décalages xshift et yshift. Ce

n’est clairement pas une solution optimale s’il faut tout décaler manuellement

à chaque fois.

Mais le principe de base est là ! Voyons maintenant comment mieux faire

avec des cas concrets.

Organigramme manuel

Nous allons continuer sur notre lancée avec un premier organigramme

dessiné manuellement. Après tout, il peut des fois être plus rapide de faire un

244

Chapitre 17. Dessiner avec TikZ

;A<

premier dessin manuellement que de chercher à tout optimiser et automatiser

dès le départ.

Le principe est très similaire à celui montré dans le code d’initiation

précédent :

1) Création de l’environnement tikzpicture et définition d’un style global pour tous les node pour garantir l’homogénéité du résultat.

2) Création du node de référence, « master ». Comme aucune coordonnée

n’est spécifiée, master.center se situe en (0,0).

3) Positionnement d’autres node (boîtes en-têtes) puis utilisation des points

d’ancrage pour placer des node en dessous des précédents (boîtes descriptives).

4) Relier les boîtes pour donner un sens à l’organigramme (\draw) et ajuster la taille si besoin avec \resizebox (package graphicx).

Concrètement, le code ressemble à :

Organigramme manuel

% Ajout au PREAMBULE

%\usepackage{graphicx, tikz}

\resizebox{\linewidth}{!}{

\begin{tikzpicture}[

every node/.style = {draw = black, fill = violet!70, line

width = 2pt, text width = 3cm, minimum width = 3.5cm,

minimum height = 1cm, text = white, align = center},

entete/.style = {font = \large},

descr/.style = {minimum height = 3cm}]

% Node de référence

\node[font = \Large] (master) {Manager};

% Autres nodes

\node[entete] (boiteA) at (-6,-3) {\'Equipe A};

\node[descr, anchor = north] at (boiteA.south) {Commercial \\

~ \\ Vente};

245

Chapitre 17. Dessiner avec TikZ

;A<

\node[entete] (boiteB) at (-2,-3) {\'Equipe B};

\node[descr, anchor = north] at (boiteB.south) {PHP \\ ~ \\

JavaScript};

\node[entete] (boiteC) at (2,-3) {\'Equipe C};

\node[descr, anchor = north] at (boiteC.south) {Support \\ ~

\\ Supervision};

\node[entete] (boiteD) at (6,-3) {\'Equipe D};

\node[descr, anchor = north] at (boiteD.south) {Analyse \& KPI

\\ ~ \\ Management};

% Tracé automatisé

\foreach \point in {A, ..., D} {\draw[ultra thick] (master.

south) --++ (0,-1cm) -| (boite\point);}

% -| <==> départ horizontal, arrivée verticale

\end{tikzpicture}

}

Manager

Équipe A

Commercial

Vente

Équipe B

PHP

JavaScript

Équipe C

Support

Supervision

Équipe D

Analyse & KPI

Management

Bon, jusque là, rien de nouveau, hormis la commande \resizebox très

pratique pour faire tenir les diagrammes quand ils ne rentrent pas dans ton

document. Pour t’en servir, c’est très simple. La syntaxe générale est la suivante :

\usepackage{graphicx}

246

Chapitre 17. Dessiner avec TikZ

;A<

\resizebox{<largeur>}{<hauteur>}{<élément>}

% Option "!" : permet de conserver les proportions

Utilisation la plus courante :

\resizebox{\linewidth}{!}{<élément>}

Si tu as besoin de plus de précisions, je te renvoie à la documentation du

package graphicx, disponible sur https://ctan.org/pkg/graphicx.

Voyons désormais un nouveau cas de diagramme. J’espère que tu aimes

la Nature.

Utilisation d’un arbre

Un arbre peut se révéler satisfaisant pour représenter un organigramme,

surtout s’il est similaire au premier cas présenté. TikZ offre la possibilité de

construire automatiquement l’arbre en question, sans avoir à te soucier du

positionnement. Au préalable, il faut penser à charger la bibliothèque TikZ

nommée trees.

Le principe de création d’un arbre est très similaire à ce que nous avons

vu jusqu’à présent :

❖ création d’un node de référence (ou la graine de l’arbre). Il s’agit alors

du niveau 0 (level 0) ;

❖ création des ramifications (child) avec la syntaxe suivante :

child { node[<options>] {<texte>} }

Pour créer de nouvelles ramifications, il faut jouer sur l’encapsulation

d’un nouveau child avant la fermeture de l’accolade finale. Il est alors

fortement recommandé d’indenter son code dans cette situation : la

relecture n’en sera que plus simple.

La ramification – ou niveau – i peut alors être appelée avec la syntaxe level i. C’est surtout utile pour définir un style propre à chaque

niveau, comme nous le verrons dans l’exemple qui va suivre ;

❖ il existe des options propres aux arbres. Par exemple, level distance

permet de définir l’espace entre chaque niveau. sibling distance fait

247

Chapitre 17. Dessiner avec TikZ

;A<

de même, mais pour les éléments d’un même niveau.

edge from parent path permet de définir la manière dont sont reliés

les éléments. Des nœuds génériques (gérés automatiquement par TikZ,

pour chaque niveau et élément) sont alors accessibles par les noms

\tikzparentnode et \tikzchildnode.

Voyons maintenant sur un cas concret ce que nous pouvons réaliser :

Création d’un arbre avec trees

% Ajout au PREAMBULE

%\usepackage{tikz}

%\usetikzlibrary{trees}

\resizebox{\linewidth}{!}{

\begin{tikzpicture}[every node/.style = {rounded corners, draw

, top color = white, bottom color = blue!20, align =

center}, level distance = 50pt, level 1/.style = {sibling

distance = 17em}, level 2/.style = {sibling distance = 8em

}, edge from parent/.style = {draw, edge from parent path

= {(\tikzparentnode.south) --+ (0,-10pt) -| (

\tikzchildnode)}}]

\node {NOM \\ Chef}

child { node {Sous-chef A}

child { node {Esclave Aa}}

child { node {Esclave Ab}}

}

child { node {Sous-chef B}

child { node {Esclave Ba}

child { node {Stagiaire Ba1}}

child { node {Stagiaire Ba2}}

child { node {Stagiaire Ba3}}

}

child { node {Esclave Bb}}

};

\end{tikzpicture}

}

248

Chapitre 17. Dessiner avec TikZ

;A<

NOM

Chef

Sous-chef A

Esclave Aa Esclave Ab

Sous-chef B

Esclave Ba

Stagiaire Ba1 Stagiaire Ba2 Stagiaire Ba3

Esclave Bb

!

Saut de ligne interdit !

Pour la création d’arbres, il est strictement interdit de sauter des

lignes pour aérer le code, sous peine d’avoir une erreur de compilation

(code mal interprété).

C’est pourquoi je recommande dans ce cas de figure d’indenter le

code, avec un décalage pour chaque niveau. C’est ce que je fais dans

mes exemples.

Avec un peu de ruse lors de la configuration de edge from parent path,

un style judicieusement configuré avec l’option grow (propre à la bibliothèque

trees), il est possible de relier différemment les éléments :

Une autre possibilité d’arbre

% Ajout au PREAMBULE

%\usepackage{tikz}

%\usetikzlibrary{trees}

\begin{center}

\begin{tikzpicture}[man/.style = {draw, fill = blue!20}, woman

/.style = {rounded corners = .8ex, draw, fill = red!20},

grandchild/.style = {grow = down, xshift = 1em, anchor =

west, edge from parent path = {(\tikzparentnode.south) |-

249

Chapitre 17. Dessiner avec TikZ

;A<

(\tikzchildnode.west)}}, first/.style = {level distance =

6ex}, second/.style = {level distance = 12ex}, third/.

style = {level distance = 18ex}, level 1/.style = {sibling

distance = 5em}]

% Parents

\coordinate

child[grow = left] {node[man, anchor = east] {Jim}}

child[grow = right] {node[woman, anchor = west] {Jane}}

child[grow = down, level distance = 0ex][edge from parent

fork down]

% Enfants & petits-enfants

child{node[man] {Alfred}

child[grandchild, first] {node[man] {Joe}}

child[grandchild, second] {node[woman] {Heather}}

child[grandchild, third] {node[woman] {Barbara}}

}

child{node[woman] {Berta}

child[grandchild, first] {node[man] {Howard}}

}

child {node[man] {Charles}}

child {node[woman] {Doris}

child[grandchild, first] {node[man] {Nick}}

child[grandchild, second] {node[woman] {Liz}}

};

\end{tikzpicture}

\end{center}

Jim Jane

Alfred

Joe

Heather

Barbara

Berta

Howard

Charles Doris

Nick

Liz

Voilà, c’est tout ce que je peux présenter concernant les arbres à l’heure

250

Chapitre 17. Dessiner avec TikZ

;A<

actuelle. Heureusement, j’ai gardé le meilleur pour la fin, avec une solution

un peu plus automatisée. Mais un interlude est nécessaire avant d’en parler.

Utiliser des flèches

Les diagrammes que j’ai présentés jusqu’à présent peuvent convenir en

l’état. Cependant, nous allons vite être limités pour transmettre plus d’informations si nous n’avons pas de flèches !

Sous TikZ, une flèche n’est ni plus ni moins qu’un trait (dessiné par

\draw) auquel des options supplémentaires sont précisées pour enrichir son

ou ses extrémités, en l’occurrence avec des flèches :

❖ -> : pour avoir une pointe de flèche à la fin du trait ;

❖ <- : pour avoir une pointe de flèche au début du trait ;

❖ <-> : pour avoir une pointe de flèche à chaque extrémité ;

❖ >-< ou ->> ou >>>- : autres combinaisons possibles et suffisamment

explicites quand tu as compris le principe de fonctionnement ;

❖ > = <fleche> : pour spécifier une autre forme de flèche que celle par

défaut. Les formes les plus classiques sont Stealth et Straight Barb.

Pour cette dernière, il faut charger la bibliothèque TikZ arrows.meta.

Et si tu veux d’autres formes, cf. la documentation officielle de TikZ

5

;

❖ et bien d’autres options, pour agrandir la taille de la flèche, colorer la

flèche d’une autre couleur que celle du trait, etc. Je renvoie, encore une

fois, à la documentation officielle de TikZ à ce sujet.

Un petit exemple d’application pour digérer tous ces éléments :

Flèches et cas pratiques

\begin{center}

\begin{tikzpicture}

\draw (0,0) -- (0,2);

\draw[->, very thick] (2,0) -- (2,2);

5. Disponible sur https://www.ctan.org/pkg/pgf.

251

Chapitre 17. Dessiner avec TikZ

;A<

\draw[<-, ultra thick] (4,0) -- (4,2);

\draw[>->, line width = 3pt] (6,0) -- (6,2);

\draw[<<-<, > = Stealth, line width = 5pt] (8,0) -- (8,2);

\end{tikzpicture}

\end{center}

Bien, finissons maintenant avec un dernier diagramme.

Un beau diagramme ?

Le principe de base reste inchangé : utilisation d’un node de référence,

positionnement des autres node par rapport à la référence ou aux nouveaux

node, utilisation judicieuse des styles pour décorer.

Si le positionnement standard est intéressant, les possibilités deviennent

plus intéressantes avec la bibliothèque positioning. Selon moi, la meilleure

façon de s’en servir est la suivante :

1) L’option node distance = <ecart> permet de définir l’écart entre

chaque node (localement sur un node ou globalement lors de l’appel

de l’environnement tikzpicture).

2) Création du node de référence, master.

3) Positionnement d’un nouveau node avec l’option : below = of master.

Résultat : celui-ci se retrouve sous master, avec une séparation de taille

<ecart>.

De manière plus générale, les 4 choix de positionnement – above, below,

left et right, qui peuvent être combinés, peuvent faire référence à un node

(ou un point d’ancrage spécifique) selon la syntaxe suivante :

<position> = of <node>

252

Chapitre 17. Dessiner avec TikZ

;A<

Et c’est tout ! Maintenant, tu es libre de faire ce que tu veux, d’automatiser toutes les distances avec des longueurs LATEX. C’est ce que je fais

en tout cas. Et je te propose ma petite template personnelle pour faire des

diagrammes :

Template pour faire un diagramme

\documentclass[a4paper, 12pt]{report}

% PDFLaTeX

\usepackage{lmodern}

\usepackage[french]{babel}

\usepackage[utf8]{inputenc}

\usepackage[T1]{fontenc}

\usepackage{enumitem, pifont}

\usepackage{pdflscape}

\usepackage[dvipsnames]{xcolor}

\usepackage{tikz}

\usetikzlibrary{arrows.meta, positioning}

\begin{document}

\everymath{\displaystyle}

\pagestyle{empty}

% Diagramme - Paramétrage

\definecolor{newblue}{RGB}{68,114,196}

\newlength{\nodesep}

\addtolength{\nodesep}{2.5cm}

\newlength{\blockwidth}

\addtolength{\blockwidth}{3.8cm}

\newlength{\blockheight}

\addtolength{\blockheight}{1.6cm}

% Style des blocs (diagramme)

253

Chapitre 17. Dessiner avec TikZ

;A<

\tikzset{

node distance = \nodesep,

block/.style = {draw = newblue, fill = newblue, text =

white, rounded corners, minimum width = \blockwidth,

minimum height = \blockheight, text width = \blockwidth -

2mm, align = center, font = \sffamily},

warning/.style = {block, draw = red, fill = red},

correct/.style = {block, draw = Green, fill = Green},

careful/.style = {block, draw = orange, fill = orange},

arrow/.style = {newblue, line width = 5pt, ->, > = Stealth

},

comment/.style = {text width = \nodesep, align = center,

font = \sffamily\itshape}

}

\begin{landscape}

\resizebox{\linewidth}{!}{

\begin{tikzpicture}

% Création des blocs

\node[block] (blocA) {Bloc A};

\node[careful, right = of blocA] (blocB) {Bloc B};

\node[block, below = of blocA] (blocC) {Bloc C};

\node[block, left = of blocC] (blocD) {Bloc D};

\node[careful, below = of blocD] (blocE) {Bloc E};

\node[careful, node distance = 2\nodesep + \blockwidth, right

= of blocE] (blocF) {Bloc F};

\node[careful, below = of blocE] (blocG) {Bloc G};

\node[careful, right = of blocG] (blocH) {Bloc H};

\node[correct, right = of blocH] (blocI) {Bloc I};

% Liens entre les blocs et commentaires

\draw[arrow] (blocA) -- (blocB);

\draw[arrow] (blocA) -- (blocC);

\draw[arrow] (blocA) -| (blocD) node[left, text width = 0.8

\blockwidth, align = left] at (blocD.west) {\textbf{Liste

:} \begin{itemize}[label = \textcolor{newblue}{\ding

{118}}, leftmargin = \*]

\item puce a

254

Chapitre 17. Dessiner avec TikZ

;A<

\item puce b\dots{} \\ ~ \\

\end{itemize}

\textbf{\color{red!75!black}$\sf\times 2$ Tada !}};

\draw[arrow, orange] (blocB) -- (blocF);

\draw[arrow] (blocC) |- (blocF);

\draw[arrow] (blocD) -- node[comment, right] {Commentaire} (

blocE);

\draw[arrow, orange] (blocE) -- (blocG);

% Création d'un node fictif (déviation)

\path (blocE) -- (blocG) coordinate [midway] (noeud fictif A);

\draw[arrow, orange] (blocF) node[comment, right] at (blocF.

east) {Commentaire \\ Commentaire} |- (noeud fictif A) -|

(blocG);

% Petite astuce (intersection / déviation)

% (blocA -| blocB) = coordonnée

\draw[arrow, orange] node[comment, left, align = right, text

width = 1.2\blockwidth] at (blocG.west) {Commentaire \\ (d

étails) \\ Commentaire \\ Commentaire} (blocG) -- (blocH);

\draw[arrow, orange] (blocH) -- (blocI);

\end{tikzpicture}}

\end{landscape}

\end{document}

Bloc A Bloc B

Bloc D Bloc C

Bloc E Bloc F

Bloc G Bloc H Bloc I

Liste :

❖ puce a

❖ puce b. . .

×2 Tada !

Commentaire

Commentaire

Commentaire

Commentaire

(détails)

Commentaire

Commentaire

255

Chapitre 17. Dessiner avec TikZ

;A<

Mon petit conseil

Avant de te lancer dans l’écriture de ton code TikZ, pose ton diagramme sur le papier. Tu verras que tu gagneras du temps et que ce

sera plus facile pour le transposer sous LATEX.

Tu peux aussi plus facilement l’améliorer sur papier en griffonnant

au lieu de réécrire 13 fois le même code parce que tu changes constamment d’avis.

Enfin, j’ai mis des noms de node en autoincrement pour faciliter

la compréhension de l’exemple. Il vaut mieux donner un nom explicite

à ton node. C’est plus facile si tu dois en ajouter un nouveau par la

suite, que de devoir décaler tous tes noms de node.

17.8 Le mot de la fin

Loin d’avoir tout expliqué sur TikZ

6

, les exemples que j’ai élaborés et

mis à disposition au sein de ce guide donnent malgré tout beaucoup d’informations et constituent une première base solide.

Naturellement, je suis loin d’être exhaustif et je me suis efforcé d’aborder

un large panel de notions que je connais. À toi désormais de faire des essais,

d’améliorer ton code et de découvrir de nouvelles possibilités !

TikZ propose un grand nombre de bibliothèques, avec des fonctionnalités

diverses et variées.

Elles sont détaillées à la partie V du guide officiel de TikZ, ainsi que sur :

http://tex.stackexchange.com/questions/42611/list-of-availabletikz-libraries-with-a-short-introduction.

Tu trouveras aussi un tableau bilan à ce sujet dans les annexes, Table A.9

p. 270.

Enfin, il existe aussi d’autres packages pour agrémenter les dessins sous

LATEX. C’est par exemple le cas de pgfornament, qui mérite le détour et

fournit des ornements intéressants.

6. Le guide officiel fait plus de 1 000 pages donc tu penses bien que je n’ai fait qu’effleurer le champ des possibles.

256

Chapitre 17. Dessiner avec TikZ

;A<

Figure 17.3 – Un premier aperçu du package pgfornament

Pour finir, je me suis amusé à réaliser un petit fond d’écran pour mon

ordinateur. Naturellement, il prône l’utilisation du LATEX et joue un peu sur

la fibre patriotique.

Il est à disposition ci-après si tu veux réaliser une capture d’écran pour

l’utiliser de ton côté. Tenter de le reproduire peut aussi constituer un bon

entraînement dans ton apprentissage de TikZ.

257

ϕ =

1 + √

5

2

LA

TE

X

LA

TE

X

Limitless creation

Good-looking reports

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras euismod fringilla felis, ac

mollis nulla pellentesque eget. Vivamus blandit porta tincidunt. Quisque ullamcorper ipsum a dui posuere, congue placerat justo vestibulum. Maecenas eleifend neque posuere gravida dignissim. Quisque quis tortor sed elit rutrum aliquam. Nulla facilisi. Mauris est sapien,

viverra vitae purus pretium, mollis posuere

neque. Duis rutrum lectus vel nunc tincidunt

condimentum. Suspendisse est sem, sodales eu

metus id, fringilla vulputate risus. Proin et ex

at nunc consectetur tempor sit amet accumsan tellus. Vestibulum quis ultricies orci. Morbi

mollis quam neque, eu vulputate libero volutpat sagittis. Phasellus scelerisque mauris id lorem viverra rhoncus. Fusce dictum velit arcu,

eget congue mi convallis pulvinar.

Chapitre 18

Faire des présentations avec

Beamer

À venir. . . (peut-être 2nd semestre 2019)

259

Annexes

260

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.1 – Les différentes possibilités de mise en forme du texte

Texte Rendu Environnement

\textbf{gras} gras bfseries

\textit{italique} italique itshape

\emph{emphase} emphase em

\textsl{penché} penché slshape

\textsc{Petites Capitales} Petites Capitales scshape

\textsf{sans empattement} sans empattement sffamily

\texttt{machine} machine (à écrire) ttfamily

261

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.2 – Liste non exhaustive des symboles disponibles sous LATEX

Code Rendu Description

\& & Esperluette

\oe et \OE œ et Œ Ligature œ

\ae et \AE æ et Æ Ligature æ

\ss ß Eszett

\no n

o Numéro

- - Tiret court

-- – Tiret moyen

--- — Tiret long

\dots . . . Points de suspension

\og et fg « et » Guillemets français ouvrants

`` (accents graves) “ Guillemets anglais ouvrants

'' (apostrophes) ” Guillemets anglais fermants

\% % Pourcent

\euro e Euro (package marvosym)

\$ $ Dollar

\textcopyright © Copyright

\textregistered ® Marque déposée

\texttrademark ™ Trademark

\# # Dièse

\{ { Accolade ouvrante

\} } Accolade fermante

\\_ \_ Underscore

\textbackslash \ Backslash

\textasciitilde ~ Tilde

262

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.3 – Liste des symboles du package pifont

32 33 ✁ 34 ✂ 35 ✃ 36 ✄ 37 ☎ 38 ✆ 39 ✇

40 ✈ 41 ✉ 42 ☛ 43 ☞ 44 ✌ 45 ✍ 46 ✎ 47 ✏

48 ✐ 49 ✑ 50 ✒ 51 ✓ 52 ✔ 53 ✕ 54 ✖ 55 ✗

56 ✘ 57 ✙ 58 ✚ 59 ✛ 60 ✜ 61 ✝ 62 ✞ 63 ✟

64 ✠ 65 ✡ 66 ✢ 67 ✣ 68 ✤ 69 ✥ 70 ✦ 71 ✧

72 ★ 73 ✩ 74 ✪ 75 ✫ 76 ✬ 77 ✭ 78 ✮ 79 ✯

80 ✰ 81 ✱ 82 ✲ 83 ✳ 84 ✴ 85 ✵ 86 ✶ 87 ✷

88 ✸ 89 ✹ 90 ✺ 91 ✻ 92 ✼ 93 ✽ 94 ✾ 95 ✿

96 ❀ 97 ❁ 98 ❂ 99 ❃ 100 ❄ 101 ❅ 102 ❆ 103 ❇

104 ❈ 105 ❉ 106 ❊ 107 ❋ 108 ● 109 ❍ 110 ■ 111 ❏

112 ❐ 113 ❑ 114 ❒ 115 ▲ 116 ▼ 117 ◆ 118 ❖ 119 ◗

120 ❘ 121 ❙ 122 ❚ 123 ❛ 124 ❜ 125 ❝ 126 ❞

161 ❡ 162 ❢ 163 ❣ 164 ❤ 165 ❥ 166 ❦ 167 ❧

168 ♣ 169 ♦ 170 ♥ 171 ♠ 172 ① 173 ② 174 ③ 175 ④

176 ⑤ 177 ⑥ 178 ⑦ 179 ⑧ 180 ⑨ 181 ⑩ 182 ❶ 183 ❷

184 ❸ 185 ❹ 186 ❺ 187 ❻ 188 ❼ 189 ❽ 190 ❾ 191 ❿

192 ➀ 193 ➁ 194 ➂ 195 ➃ 196 ➄ 197 ➅ 198 ➆ 199 ➇

200 ➈ 201 ➉ 202 ➊ 203 ➋ 204 ➌ 205 ➍ 206 ➎ 207 ➏

208 ➐ 209 ➑ 210 ➒ 211 ➓ 212 ➔ 213 → 214 ↔ 215 ↕

216 ➘ 217 ➙ 218 ➚ 219 ➛ 220 ➜ 221 ➝ 222 ➞ 223 ➟

224 ➠ 225 ➡ 226 ➢ 227 ➣ 228 ➤ 229 ➥ 230 ➦ 231 ➧

232 ➨ 233 ➩ 234 ➪ 235 ➫ 236 ➬ 237 ➭ 238 ➮ 239 ➯

241 ➱ 242 ➲ 243 ➳ 244 ➴ 245 ➵ 246 ➶ 247 ➷

248 ➸ 249 ➹ 250 ➺ 251 ➻ 252 ➼ 253 ➽ 254 ➾

263

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.4 – Liste non exhaustive des polices sous LATEX

Police Package <code-police>

Bookman bookman pbk

Charter charter bch

Computer Modern Roman (défaut) cmr

Computer Modern Sans Serif (défaut) cmss

Computer Modern Typewriter (défaut) cmtt

Courier courier pcr

Fourier fourier put

Garamond ebgaramond

ebgaramond-maths

Helvetica helvet phv

Latin Modern Roman lmodern lmr

Latin Modern Sans Serif lmodern lmss

Latin Modern Typewriter lmodern lmtt

New Century Schoolbook newcent

Palatino mathpazo

TEX Gyre Bonum tgbonum qbk

Times mathptmx ptm

Zapf Chancery chancery

264

Chapitre 18. Faire des présentations avec Beamer

;A<

265

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.5 – Liste non exhaustive des symboles mathématiques disponibles

sous LATEX

Code Rendu Description

$i\_2$ i2 Indice

$i^3$ i

3 Exposant

$\frac{a}{b}$ a

b

Fraction

$\cfrac{a}{b + \cfrac{c}{d}}$

a

b +

c

d

Fraction (étages)

$\times$ × Multiplication

$\pm$ ± Plus ou moins

$\leq$ et $\geq$ ≤ et ≥ Inégalités larges

$\leqslant$ et $\geqslant$ ⩽ et ⩾ Inégalités larges (bis)

$\equiv$ ≡ Congruence

$\neq$ ̸= Non égal

$\simeq$ ≃ Environ égal

$\approx$ ≈ Environ égal (bis)

$\sim$ ∼ Équivalence

$\forall$ ∀ Pour tout élément

$\exists$ ∃ Existence

$\Rightarrow$ ⇒ Implication

$\infty$ ∞ Infini

$\int$ Z

Intégrale simple

$\iint$ ZZ Intégrale double

$iiint$ ZZZ Intégrale triple

$\oint$ I

Intégrale curviligne

$\int\_0^{+\infty} f(x)\,dx$ Z +∞

0

f(x) dx Intégration

$\sum$ X Somme

$\partial$ ∂ Dérivée partielle

266

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.6 – La liste complète des lettres grecques sous LATEX

α \alpha η \eta ξ \xi τ \tau

β \beta θ \theta π \pi υ \upsilon

γ \gamma ϑ \vartheta ϖ \varpi ϕ \phi

δ \delta κ \kappa ρ \rho φ \varphi

ϵ \epsilon λ \lambda ϱ \varrho χ \chi

ε \varepsilon µ \mu σ \sigma ψ \psi

ζ \zeta ν \nu ς \varsigma ω \omega

Γ \Gamma Λ \Lambda Σ \Sigma Ψ \Psi

∆ \Delta Ξ \Xi Υ \Upsilon Ω \Omega

Θ \Theta Π \Pi Φ \Phi

267

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.7 – Liste non exhaustive des commandes et options disponibles

avec TikZ

Commandes / Options Description

-- Pour tracer un trait droit

|- ou -| Pour tracer un trait vertical

puis horizontal (ou l’inverse)

|| Indisponible !

rectangle Choix de forme

(contour fermé) circle

cycle Pour fermer un chemin

plot Pour tracer une fonction (maths)

\draw[<options>] (A) -- (B); Tracer un trait entre A et B

\fill[<options>] (A) -- (B); Remplir la zone entre A et B

\filldraw[<options>] (A) -- (B); Tracer et remplir la zone entre A et B

draw = <couleur> Définir la couleur du tracé

fill = <couleur> Définir la couleur du remplissage

line width = <longueur> Définir l’épaisseur du trait

dotted ou dashed Changer la forme du trait

rounded corners

Changer les jointures

sharp corners

line join = <type> Autres jointures

(round, bevel & miter)

shift = {(<coord>)} Décalage

xshift = <taille> Décalage horizontal

yshift = <taille> Décalage vertical

\path[<options>] (A) -- (B); Définir un chemin entre A et B

\coordinate (<nom>) at (<coord>); Définir une coordonnée <nom>

au point (<coord>)

node cf. Table A.8

268

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.8 – Liste non exhaustive des options disponibles avec TikZ (node)

\node[<options>] (<nom>) at (<coord>) {<texte>};

Placer <texte> à l’emplacement (<coord>) (autre node ou coordonnées) et lui attribuer le nom <nom> (optionnel)

Options Description

draw = <couleur> Afficher le contour du node

fill = <couleur> Remplir l’intérieur du node

text = <couleur> Colorer le texte à l’intérieur du node

font = <mise-en-forme> Mettre en forme le texte du node

(\bfseries, itshape, \small, etc.)

align = <position> Alignement horizontal du texte

à l’intérieur du node

(left, center ou right)

inner sep = <taille> Espacement entre le texte et

le bord du node

outer sep = <taille> Espacement entre le bord du node

et les autres éléments

text width = <taille> Largeur du bloc de texte du node

minimum width = <taille> Largeur minimale du node

minimum height = <taille> Hauteur minimale du node

<position> Indiquer la position du node

par rapport à (<coord>)

(above, below, left ou right)

anchor = <ancre> Indiquer le point d’ancrage de (<coord>)

rotate = <angle> Faire pivoter le node

Autres solutions de positionnement : cf. la bibliothèque positioning.

269

Chapitre 18. Faire des présentations avec Beamer

;A<

Table A.9 – Liste des librairies disponibles avec TikZ

\usetikzlibrary{<nom-bibliotheque>}

Bibliothèque TikZ Description

3d Dessiner des formes en 3D

angles Dessiner des angles

animations Créer des animations

arrows.meta Obtenir plus de flèches et pouvoir

en créer

automata Dessiner des automates finis (diagrammes d’état) et des machines

de Turing

babel Interagir avec le package éponyme

et éviter les conflits

backgrounds Créer des arrières-plans colorés et

afficher un quadrillage

bending Courber les flèches

calc Calculer des coordonnées

calendar Créer des calendriers

chains Créer des chaînes

circuits

Dessiner des circuits électriques

(package circuitikz si besoin)

circuits.ee

circuits.logic

circuits.logic.CDH

circuits.logic.IEC

circuits.logic.US

circuit.ee.IEC

circular Créer des dessins algorithmiques

circulaires

(suite sur la page suivante)

270

Chapitre 18. Faire des présentations avec Beamer

;A<

Bibliothèque TikZ Description

curvilinear Dessiner des transformations non

linéaires (comme des courbes de

Bézier)

datavisualization

Définir les styles nécessaires à la

visualisation de données datavisualization.formats.functions

datavisualization.polar

decorations

Appliquer des transformations à

des chemins (path) et les décorer

decorations.footprints

decorations.fractals

decorations.markings

decorations.pathmorphing

decorations.pathreplacing

decorations.shapes

decorations.text

er (entity-relationship) Dessiner des diagrammes entitéassociation

external Exporter des dessins TikZ

fadings Estomper les couleurs

fit Créer un node qui contient un jeu

de coordonnées

fixedpointarithmetic Augmenter la limite de calcul

de TikZ (package fp obligatoire,

temps de calcul plus longs)

folding Créer des patrons ou des objets à

plier

force Dessiner avec des “efforts” (node

tirés ou comprimés)

fpu (floating point unit) Permettre le calcul scientifique

avec des nombres réels

graphdrawing (LuaLATEX) Automatiser le dessin de graphes

graphs et graphs.standard Dessiner des graphes avec la commande graph

(suite sur la page suivante)

271

Chapitre 18. Faire des présentations avec Beamer

;A<

Bibliothèque TikZ Description

intersections Calculer la (les) intersection(s) de

2 chemins

layered Tracer des graphes avec des

couches horizontales

lindenmayersystems Créer des “L-systems”

math Définir des fonctions mathématiques et exécuter des opérations

mathématiques

matrix Définir des options et styles supplémentaires pour les matrices de

node

mindmap Créer des cartes heuristiques

patterns Définir de nouveaux motifs de remplissage

perspective Dessiner des objets en perspective

avec 1, 2 ou 3 points de fuite

petri Dessiner des réseaux de Petri

phylogenetics Tracer des arbres phylogénétiques

plothandlers Définir des « plot handlers » supplémentaires

plotmarks Définir des marqueurs (sommets

d’un tracé)

positioning Définir des options supplémentaires pour placer les node

profiler Simplifier l’optimisation de la

compilation

quotes Placer des commentaires facilement à côté d’un node

rdf (resource description framework)

Ajouter des “annotations” à un fichier généré par TikZ

routing Connecter les extrémités d’un

graphe

(suite sur la page suivante)

272

Chapitre 18. Faire des présentations avec Beamer

;A<

Bibliothèque TikZ Description

scopes Simplifier l’appel de l’environnement scope

shadings Créer des dégradés de couleurs

shadows Créer des ombrages

shapes.arrows

Définir de nouveaux formats de

node et plus encore

shapes.callout

shapes.gates.ee

shapes.gates.ee.IEC

shapes.gates.logic

shapes.gates.logic.IEC

shapes.gates.logic.US

shapes.geometric

shapes.misc

shapes.multipart

shapes.symbols

spy Faire des grossissements

svg.path Définir des chemins avec la syntaxe

SVG

through Créer des formes qui passent par

un point précis

topaths (chargé automatiquement)

Définir l’opérateur to

trees Dessiner des arbres

turtle Créer des “turtle graphics” (langage Logo)

views Créer des “vues” (transformations

d’une partie d’un dessin)

Fin du tableau

273