Exposés sur LaTeX

Cours 2

Les fondamentaux : le document source

Thierry Masson

Centre de Physique Théorique
Campus de Luminy, Marseille

Cours 2 – Les fondamentaux : le document source

- Le document source : les bases
- Structure du document source
- Repères sur les concepts de programmation
- Tour d'horizon de commandes de base

« Mon pauvre oncle disait souvent :
"Il faut toujours tourner sa langue sept fois
dans sa bouche avant de parler."
Que devrait-on faire avant d'écrire ? »
Gérard de Nerval
Les illuminés

Le document source : les bases

Où l'on prend contact avec les rudiments d'un langage cabalistique grâce auquel des incantations magiques feront de nous les maîtres de la page blanche…

La philosophie de TEX

Avec TEX et LATEX, on tape du texte (presque) normalement et on fait appel de temps en temps à des macros (des morceaux de programmes) pour obtenir des effets divers ou des fonctionnalités particulières.

⚠ Il y a des caractères réservés pour la syntaxe des macros.
En voici la liste, ainsi que la façon de les obtenir concrètement dans du texte :

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

(voir aussi le *package* **textcomp**)

Avec des macros on peut : modifier les styles typographiques (taille, graisse, forme, couleur…), modifier la mise en page (marges, indentations…), ajouter des notes en bas de page, composer des titres, générer une table des matières, composer des mathématiques (symboles, disposition des formules…), insérer des images, créer des tableaux, composer des dessins…

Le fait que TEX soit avant tout un langage de programmation permet de créer des macros capables de remplir de très nombreuses tâches.

Parmi ces macros, figurent les “commandes”, sous la forme `\cmd` (“\” suivi d’un nom).
TEX ne connaît que des commandes avec ou sans arguments.
LATEX a introduit les arguments optionnels et la notion d’environnement.

Le rôle des caractères réservés

Les caractères réservés se rencontrent dans les situations suivantes :

- `\` marque le début d'une macro.
  Exemple : `\newpage` est la commande pour démarrer une nouvelle page.
- `{` et `}` ouvrent et ferment un **groupe**, c'est à dire une entité unique pour TEX.
  Cela sert à créer un contexte local ou à englober l'argument (long) d'une macro.
  Exemples : `{\large texte en grand}`, `\textit{texte en italique}`.
  ➟ les (re)définitions opérées à l'intérieur d'un groupe restent locales.
- `%` marque le début d'un commentaire.
- `~` crée un espace insécable.
- `$` commence et termine le mode mathématique.
- Dans le mode mathématique, `^` sert à placer des exposants et `_` sert à placer des indices.
- `&` intervient dans les séparations des colonnes de tableaux et de matrices, et comme positions d'alignement de formules de mathématique.
- `#` se rencontre dans la définition de macros.

⚠ Les caractères `[` et `]` ne sont pas réservés, mais ils interviennent pour délimiter des arguments optionnels dans des macros LATEX, ce qui fait que leur rôle est ambigu.
➟ cette situation génère parfois des erreurs incompréhensibles.

Commandes et environnements, généralités

Il y a 4 types de commandes :

1. `\cmd` : sans aucun argument ;
2. `\cmd{-}` : un ou plusieurs arguments obligatoires, chacun entre accolades `{-}` ;
3. `\cmd[-]` : un ou plusieurs arguments optionnels, chacun entre crochets `[-]` ;
4. `\cmd[-]{-}` : un ou plusieurs arguments optionnels et un ou plusieurs arguments obligatoires.

Les environnements sont des macros qui englobent un morceau de texte.
De ce fait, un environnement travaille toujours avec du contenu : le texte englobé.
Comme pour les commandes, il y a 4 types d'environnements :

1. `\begin{env} ... \end{env}` : sans argument ;
2. `\begin{env}{-} ... \end{env}` : un ou plusieurs arguments obligatoires ;
3. `\begin{env}[-] ... \end{env}` : un ou plusieurs arguments optionnels ;
4. `\begin{env}[-]{-} ... \end{env}` : un ou plusieurs arguments optionnels et un ou plusieurs arguments obligatoires.

La place des arguments optionnels est souvent avant les arguments obligatoires, mais pas forcément.

Commandes et environnements, exemples

Exemples de commandes :

\documentclass[-]{-}, \usepackage[-]{-}
\chapter[-]{-}, \section[-]{-}, \footnote{-}, \tableofcontents
\Large, \tiny, \textbf{-}, \slshape, \textcolor{-}{-}
\par, \bigskip, \medskip, \smallskip, \vspace{-}
\newpage, \clearpage, \pagebreak[-]
\alpha, \otimes, \sum, \sin, \sqrt{-}, \frac{-}{-}
\includegraphics[-]{-}, \scalebox{-}{-}
\{, \$, \,, \\

Exemples d'environnements :

\begin{document} ... \end{document}
\begin{enumerate} ... \end{enumerate}
\begin{abstract} ... \end{abstract}
\begin{flushleft} ... \end{flushleft}
\begin{tabular}{-} ... \end{tabular}
\begin{minipage}[-]{-} ... \end{minipage}
\begin{equation} ... \end{equation}
\begin{figure}[-] ... \end{figure}
\begin{thebibliography}[-] ... \end{thebibliography}

Commandes et environnements, subtilités

Certaines commandes et certains environnements semblent équivalents :
`\begin{center} ... \end{center}`, `{\centering ... }`, `\centerline{-}`.
En réalité, il y a des différences : l'environnement `center` ajoute des espaces avant et après le texte centré contrairement à `\centering`, et ces deux macros peuvent contenir des lignes interrompues par `\\` alors que `\centerline{-}` ne le peut pas.
(`\centerline{-}` est du pur TEX, qu'il faut bannir en LATEX.)

Certaines commandes ne sont définies que localement.
`\item` n'est défini que dans les environnements `itemize`, `description` et `enumerate`.
Les commandes mathématiques ne sont définies que dans le mode mathématique :
`$ ... $` ou dans les environnements `equation`, `align`, `gather`, `multline`…

Certaines commandes représentent des morceaux de texte ou des longueurs : ce sont donc plus des variables au sens de la programmation que des macros.
Par contre, les "variables" compteurs ne sont pas des commandes mais juste des noms.
⚠ Risques fréquents de confusions…

Il est facile de définir soi-même des commandes et des environnements, LATEX fournit les outils (les commandes !) pour ça.

Remarque technique (mais parfois utile).
Un environnement de type `\begin{`*env*`} ... \end{`*env*`}` est en réalité un ensemble de deux commandes : `\env` (avec d'éventuels arguments et options) qui ouvre et `\endenv` qui ferme.

Le B.A.BA de la frappe sous LaTeX

- TeX ignore ce qui est sur la même ligne au delà d'un caractère %.
  ➟ permet de placer des commentaires dans le fichier source et de désactiver des lignes de texte.
  ⚠ Le "retour chariot" à la fin d'une ligne contenant un % est aussi ignoré par TeX.
- TeX ignore les retours chariot isolés (un seul passage à la ligne).
  ➟ on peut découper un paragraphe dans le code source en lignes pas trop longues.
  Typographie : un retour chariot isolé est considéré comme un espace, donc on reste dans le paragraphe courant.
- TeX traite les tabulations comme des espaces.
  ➟ on peut utiliser des tabulations pour structurer le fichier source.
  Typographie : il y a un environnement spécifique pour créer des tabulations.
- Pour TeX, un ou plusieurs espaces entre mots signifie la même chose.
  ➟ rappel : une tabulation et un retour chariot (isolé) sont aussi des espaces !
  Typographie : pas "d'espaces" superflus entre les mots, si on veut augmenter l'espace entre deux mots, il faut utiliser une commande, par exemple `\hspace{`⟨*dim*⟩`}`.
- TeX a tendance à absorber les espaces qui suivent une commande :
  `\LaTeX est formidable` ➟ LaTeXest formidable,
  `\LaTeX{} est formidable` ➟ LaTeX est formidable.

Le B.A.BA de la frappe sous LaTeX (suite)

- `\\` interrompt la ligne courante et passe à la ligne.
  ➡ à utiliser surtout avec des environnements `center`, `flushleft`, `flushright`…
  Typographie : *la nouvelle ligne ne commence pas un nouveau paragraphe.*
  `\\[`⟨*dim*⟩`]` ajoute en plus la longueur ⟨*dim*⟩ verticalement entre les lignes.
  `\\*[`⟨*dim*⟩`]` interdit en plus un saut de page.
  ➡ syntaxes valables aussi bien dans le texte qu'en mathématique.
- Une ligne vide (ou plus) signifie "passage dans un nouveau paragraphe".
  ➡ à user sans modération pour aérer le code source.
  Typographie : arrêt du paragraphe courant, passage à la ligne avec éventuel espace vertical supplémentaire, indentation de la nouvelle ligne…
  Equivaut à placer `\par` à la fin de la ligne du paragraphe à terminer.
- Un `\␣` (`\` suivi d'un espace) créé un espace de largeur fixe.
  Un `~` collant deux mots crée un espace de largeur fixe et insécable.
  Typographie : les espaces habituels entre les mots ont une largeur variable pour permettre la justification des lignes.
- Du code encadré par le signe `$` correspond au mode mathématique dans le texte.
  ➡ c'est ce qu'on appelle des mathématiques *textstyle*.
  Il y a le mode *displaystyle* qui place les formules hors des paragraphes.
  Typographie : la forme (*shape*) passe à "italique", les espacements entre lettres sont constants, certaines macros sont activées (`_` et `^`), les accents de texte sont interdits…

Conseils élémentaires pour maîtriser LaTeX

Apprendre LaTeX, c'est apprendre le language TeX/LaTeX ainsi qu'un logiciel d'interface.

Règle 1 : Si votre clavier n'est pas américain, *trouver comment produire le symbole* `\`.
Règle 2 : Rechercher un symbole inconnu dans **Comprehensive LaTeX symbol list**.
Règle 3 : Rechercher la syntaxe d'une macro dans **LaTeX Reference Manual**.

L'interface : C'est le lieu de travail, donc l'essentiel.

- Choisir un environnement de travail riche, puissant et surtout bien maitrisé.
- Créer un ensemble de fichiers d'exemples (vides de contenu sémantique) pour différents type de documents : articles, lettres, mémoires, rapports, CV…
  Certains logiciels gèrent de tels modèles qu'il est facile d'appeler lors de la création d'un nouveau document.

Gestion du code source : Le code source, c'est le reflet de notre personnalité…

- Aérer le texte source, mettre des lignes de commentaires et des indentations.
  Pour LaTeX, une ou plusieurs lignes vides, c'est la même chose.
- Marquer clairement le début et la fin des groupes
  (`{ ... }` ou `\begin{env} ... \end{env}`).
- S'il s'agit d'un gros projet (rapport, livre), découper le texte en petits morceaux, et définir un fichier "maître" qui appelle les autres avec des `\include{-}`.
  ➡ Chaque fichier de travail (chapitre par exemple) est plus petit.
  ➡ Commenter certains `\include{-}` pour ne pas compiler tout le document.

Conseils élémentaires pour maîtriser LaTeX (suite)

Gestion des erreurs : Les erreurs sont inévitables, il faut savoir les gérer.

- À l'ouverture d'un groupe , le fermer immédiatement avant de le remplir.
- Compiler souvent le texte pour éviter d'accumuler des erreurs et pour pouvoir mieux cerner les zones en cause.
  ⚠ Une erreur à un endroit peut se révéler dans la compilation bien plus loin !
- Observer la console de compilation, et si ça ne suffit pas, ouvrir le fichier .log. C'est une mine d'informations qui permet de suivre le déroulement de la compilation (plus d'infos dans le .log que dans la console) et de comprendre la source des erreurs.
- En cas de problème, déplacer le `\end{document}` vers le haut jusqu'à trouver la zone qui ne compile pas ou mettre en commentaires de grands blocs de texte (certains logiciels le permettent facilement).

Personnalisation : Ne pas hésiter à personnaliser votre façon de travailler.

- Définir des macros sémantiques, qui auront un sens, et dont il sera possible de modifier le comportement jusqu'à la dernière compilation.
  `\section{-}` est une macro sémantique qu'il est possible de modifier.
  Dans cette présentation : `\dossier{texmf}` produit texmf et `\pkge{geometry}` produit **geometry**…
- Ne pas hésiter à utiliser un *package* qui fera le travail demandé…

De l'auteur à l'imprimeur

En tant qu'auteurs, nous sommes les premiers maillons d'une chaîne de production qui se destine en général à produire un document imprimé.
Où s'arrête notre activité dans cette chaîne ?
Il y a en gros deux types de situations.

1. L'auteur rend sa copie à un éditeur qui finalise la production.
   ➟ article de recherche, chapitre d'un livre collectif…
2. L'auteur contrôle jusqu'au bout l'aspect final du document.
   ➟ cours distribué au format PDF, rapport d'activité…

Dans le premier cas, l'auteur doit fournir le code source LaTeX :

- Ce code source doit rester le plus portable possible.
  ➟ N'utiliser que des *packages* installés par défaut (CTAN).
- Employer la feuille de style de l'éditeur si elle existe.
- Respecter au mieux la séparation du fond et de la forme.
  ➟ Éviter de jalonner le texte de commandes de mise en forme.

Dans le second cas, l'auteur doit seulement rendre un fichier PDF.
Il reste totalement maître du code source, de la façon de le gérer et de l'organiser, et concrètement de la nature des commandes et des *packages* utilisés.

Structure du document source

Où l'on découvre enfin le vrai visage d'un document source, et où l'on s'instruit de la diversité des classes et des modules…

Structure LaTeX du document source

Tout fichier LaTeX a la structure suivante :

`\documentclass[...]{...}`	Choix de la classe du document.
`\usepackage[...]{...}`	
`...`	
`\usepackage[...]{...}`	Appel des *packages*…
`AUTRES DÉCLARATIONS`	… et des définitions générales globales.
`\begin{document}`	
`CORPS DU TEXTE`	Le corps du texte est contenu dans l'environnement `\begin{document}` ... `\end{document}`.
`\end{document}`	

Tout ce qui est au delà de `\end{document}` est ignoré par LaTeX.

On appelle **préambule** la zone comprise entre `\documentclass[-]{-}` et `\begin{document}`.

`\documentclass[-]{-}` : les classes

La classe d'un document détermine plusieurs points importants :

- son apparence ;
- sa structure logique.

Les classes standard de LaTeX :

`article` C'est la classe usuelle pour les articles scientifiques.
Des déclinaisons conçues pour des revues particulières existent : **`amsart`** et **`amsproc`** (American Mathematical Society), **`revtex`** (American Institute of Physics), **`svjour`** (Springer Verlag Journals)…
Ces déclinaisons sont plus riches en métadonnées sur les auteurs et la nature du document.
Les options de ces déclinaisons permettent de sélectionner le journal final, l'état *draft* ou *referee* (double interlignes)…

`report` C'est une classe prévue pour des documents plus conséquents avec chapitres et résumé. Peut convenir pour un rapport d'étudiant.

`book` Comme son nom l'indique, est spécialement prévue pour les livres (pas de résumé).
En fait, **`report`** peut aussi faire l'affaire pour des livres.
L'AMS fournit une déclinaison **`amsbook`** pour ses collections.

\documentclass[-]{-} : les classes (suite)

Autres classes :

beamer C'est une classe fort riche qui permet des présentations par vidéo-projecteurs.
C'est celle utilisée par ce document.
Nombreuses fonctionnalités : système de "pauses" pour faire apparaître le texte graduellement, environnements divers pour mise en valeur, gestion de thèmes (aspect général, éléments présents, couleurs…), préparation d'une version "article" de l'exposé…
À consommer sans modération !

memoir Cette classe a été conçue par son auteur pour répondre à l'éternelle question de l'utilisateur usuel de LaTeX : comment modifier l'aspect de tel ou tel élément du document ?
Cette classe est basée à l'origine sur **book**, qu'elle étend de nombreuses fonctionnalités.
C'est une solution "tout-en-un" à de nombreux problèmes, plutôt qu'une solution "modulaire" qui consisterait à ajouter une fonctionnalité grâce à un *package* approprié.
De nombreux *packages* sont donc déjà "intégrés" dans **memoir**, sans risque d'incompatibilité.
Le prix à payer : la taille du projet et de sa documentation !

`\documentclass[-]{-}` : les classes (suite)

Autres classes :

Les classes de KOMA-script C'est un ensemble de classes et de *packages* associés qui remplacent les classes standard de LaTeX : `scrbook`, `scrreprt`, `scrartcl`, `scrlttr2`.
L'idée est de fournir à l'utilisateur des commandes de personnalisation et un meilleur contrôle du style du document.

Autres Il existe d'autres classes plus confidentielles ou vouées à disparaître : `slides` remplacée par `beamer` ; `letter` qui n'est pas très efficace…

Dans ce cours, il sera essentiellement question des classes usuelles.
Les classes de KOMA-script et `memoir` seront ignorées.

`\documentclass[-]{-}` : les options de classes

De nombreuses options modifient l'aspect du document :

Taille des caractères : `10pt`, `11pt`, `12pt`.

Taille de la page : `a4paper`, `a5paper`, `letterpaper`…

Mise en page : `twoside`, `landscape`, `onecolumn`, `twocolumn`, `openright`, `openany`, `titlepage`.

Divers : `draft`, `final`, `leqno`, `fleqn`, `openbib`.

Certaines de ces options sont surclassées par des *packages* qui font mieux :
`geometry` permet de gérer la taille de la page et son orientation,
`multicol` gère les mises en page multi-colonnes,
`amsmath` propose des options `fleqn`, `leqno` et `reqno`,
`titling` personnalise la page de titre,
`natbib` redéfinit la gestion de la bibliographie,
`draftwatermark` s'occupe de signaler que le document est un "draft"…

Les classes proposées par les éditeurs de revues, **`amsart`**, **`revtex`** et **`svjour`**… proposent d'autres options personnalisées.

Certaines options proposées par les classes sont utilisées (et fournies) par des *packages* :
`draft` et `final` sont proposées dans **`graphicx`** et **`hyperref`**, `a4paper`, `a5paper`, `letterpaper`… sont disponibles dans **`geometry`**…

`\usepackage[-]{-}` : les *packages*

Les *packages* sont des modules qui ajoutent des fonctionnalités à LaTeX.
➟ il faut donc savoir si un *package* fait ce qu'on désire, et lequel.

Un *package* contient un ensemble de macros dans un fichier .sty (ou plusieurs fichiers).
Une fois appelé par la commande `\usepackage{-}`, on a accès à des commandes et des environnements nouveaux.
On peut appeler plusieurs *packages* en même temps :
`\usepackage{pack1,pack2,...}`
➟ regrouper les *packages* thématiquement proches.

Il y a des *packages* pour des usages très différents : encodage du texte source, gestion des polices, gestion de l'apparence des pages, inclusion de graphiques, fonctionnalités mathématiques avancées, gestion de langues, dessins internes au document, programmation…

Les *packages* peuvent comporter des options qu'on peut activer lors de l'appel.
La commande `\usepackage[english,french]{babel}` appelle le *package* **`babel`** avec les options `english` et `french`.

\usepackage[-]{-} : les *packages* (suite)

⚠ *Tous les packages sont égaux, mais certains sont plus égaux que d'autres.*

- Certains *packages* modifient des fonctionnalités fournies par d'autres *packages*
  ➡ il est préférable de les appeler en dernier (**babel**, **hyperref**…).
- Des *packages* différents peuvent avoir des fonctionnalités semblables et entrer en conflit l'un avec l'autre.
- *Un package peut en cacher un autre* : il est possible qu'un *package* appelle d'autres *packages*, donc surveiller les incompatibilités induites.

Pour certains projets, on peut créer soi-même un *package* qui fera appel à d'autres *packages* et qui ajoutera certaines macros personnelles.

Des livres sur LaTeX permettent de découvrir l'existence de *packages* intéressants et d'en comprendre le fonctionnement.
Le site du CTAN permet de chercher dans la description des *packages*.

Classes, styles, conflicts: The biological realm of LaTeX, par D. Verna.

Structure logique et sémantique d'un document

Chaque classe a sa propre structure logique.

La structure logique la plus utilisée est celle des découpages en parties, chapitres, sections, sous sections, paragraphes et sous paragraphes.
Les classes **report** et **book** définissent les commandes `\part`, `\chapter`, `\section`, `\subsection`, `\subsubsection`, `\paragraph`, `\subparagraph`.
La classe **article** ne connaît pas `\chapter`.

Ces commandes acceptent un argument optionnel pour donner une version courte.

```
\section[Titre court]{Version longue du titre qui n'en finit pas}
```

Le titre long est celui utilisé dans le corps du texte.
Le titre court est utilisé à la fois dans la table des matières et dans les entêtes des pages.

Après le texte principal, on peut placer des annexes avec la commande `\appendix`.
Automatiquement la numérotation de plus haut niveau (`\section` pour **article**, `\chapter` pour **report** et **book**) devient alphabétique.

La classe **book** permet de diviser le texte en trois grandes parties :
une préface qui démarre avec `\frontmatter`, le corps du texte qui démarre avec `\mainmatter` et une post-face qui démarre avec `\backmatter`.
Les chapitres ne sont pas numérotés dans la préface et la post-face.
La numérotation des pages se fait en romain dans la préface.

Éléments gérés par LaTeX

LaTeX gère un certain nombre d'éléments automatiquement.

Numérotations : LaTeX s'occupe de numéroter de nombreux éléments : parties, chapitres,…, paragraphes, pages, notes de bas de page, tableaux, figures, équations, théorèmes (ou équivalents), items de listes…

Table des matières : En se basant sur les commandes `\chapter{-}`, `\section{-}`,… la commande `\tableofcontents` imprime une table des matières où elle est placée. Selon la classe du document, cette table des matières démarre un nouveau chapitre (`report` et `book`) ou une section (`article`).
Il y a des commandes équivalentes pour la liste des tableaux (`\listoftables`) et la liste des figures (`\listoffigures`).

Références croisées : La commande `\label{`*lbl*`}` permet de faire référence ailleurs dans le document au numéro actuel (qui dépend du contexte : chapitre, section, note de bas de page, équation, tableau, figure, théorème, item…) et à la page actuelle.
`\ref{`*lbl*`}` imprime le numéro ainsi préservé,
`\pageref{`*lbl*`}` imprime la page.
lbl est interne au document et ne doit pas être utilisé deux fois dans `\label{-}`.

Bibliographie, Index : La gestion de la bibliographie peut être largement automatisée grâce à `bibtex`, et la gestion des index est réalisée grâce à `makeindex`.
Plus de détails seront donnés par la suite.

Les multiples compilations

Pour réduire l'utilisation de la mémoire vive, TEX (et donc LATEX) ne conserve en mémoire aucune donnée sur le texte lors d'une compilation.
➡ LATEX utilise des fichiers auxiliaires dans lesquels il place des données à préserver : .aux par défaut, mais des *packages* en produisent d'autres.
Ces fichiers auxiliaires sont alors utilisés pour formater des fichiers "prêts à l'emploi" : .toc, .lof, .lot, .bbl, .ind…

- Première étape : en compilant, LATEX remplit de données les fichiers auxiliaires.
  ➡ .aux contient les données pour les références croisées, les entrées de la table des matières, les références bibliographiques utilisées…
- Certains moteurs doivent traiter certaines données : `bibtex` extrait les entrées bibliographiques d'une base de donnée .bib et produit un .bbl, `makeindex` formate les entrées de l'index sauvegardées dans .idx et produit un .ind…
  ➡ lancer ces moteurs si LATEX a modifié les données de bibliographie ou d'index.
- Compilation suivante : LATEX lit et met à jour les fichiers auxiliaires, utilise les fichiers "prêts à l'emploi" (.toc, .bbl, .ind…).
  ⚠ LATEX s'occupe lui même de créer le fichier formaté .toc (table des matières) à partir du .aux ➡ il faut parfois 3 compilations pour stabiliser la table des matières !

Produire un gros document exige plusieurs compilations LATEX, entrecoupées de l'usage de moteurs tels que `bibtex` et `makeindex`.
La console de compilation indique si le fichier .aux a été modifié.

Découper un document en pièces

Lorsqu'on gère un projet important, il est utile de le découper en pièces plus petites.

De façon générale, le document source peut demander à insérer d'autres documents.

`\input{fichier.tex}` lit le fichier fichier.tex à l'endroit désigné, l'interprète, et retourne ensuite à la suite du document courant.
➟ n'importe où dans le document (préambule ou corps du texte) ;
➟ on peut ainsi inclure une portion du texte final ou un ensemble de commandes ;
➟ la recherche se fait dans les dossiers texmf comme décrit auparavant : le fichier n'est pas obligé d'être dans le dossier courant.

`\include{chap1}` *dans le corps du texte* inclut le fichier chap1.tex (⚠ extension). Cette commande démarre une nouvelle page avant et après.
➟ réserver à des chapitres entiers de documents importants (livres, polycopiés…) ;
➟ un fichier .aux est créé pour chaque fichier inclus.

`\includeonly{chap1,chap3,chap4}` *dans le préambule* permet de sélectionner certains fichiers seulement.

Repères sur les concepts de programmation

Où l'on commence à découvrir la grammaire parfois obscure de notre langage cabalistique…

Définir une commande de base

Avant même de découvrir des commandes, on va apprendre à en définir.
LaTeX fournit la commande `\newcommand{-}{-}` : le premier argument est le nom de la commande à définir ; le second argument est le code définissant cette commande.
Le code peut être (presque) n'importe quoi.
➡ on peut y stocker du texte tout bête ou produire de grandes choses !

```
\newcommand{\qed}{Quod erat demonstrandum}
```

Cette commande produit "Quod erat demonstrandum" lorsqu'elle est invoquée.

```
\newcommand{\formuleEinstein}{Formule d'Einstein : $E=mc^2$}
```

`\formuleEinstein` ➡ Formule d'Einstein : $E = mc^2$

Pour définir une commande avec arguments, on utilise la syntaxe :
`\newcommand{-}[-]{-}` où `[-]` contient le nombre d'arguments.
Une commande ne peut pas accepter plus de 9 arguments.

```
\newcommand{\xpy}[2]{$#1$ à la puissance $#2$ s'écrit $#1^#2$}
```

`\xpy{3}{5}` ➡ 3 à la puissance 5 s'écrit 3^5
`\xpy{(a+b)}{n}` ➡ $(a+b)$ à la puissance n s'écrit $(a+b)^n$

Si une commande est déjà définie, on peut la redéfinir en utilisant `\renewcommand` à la place de `\newcommand`, avec la même syntaxe.

La création de commandes et d'environnements fera l'objet d'un autre cours (plus long).

Les dimensions

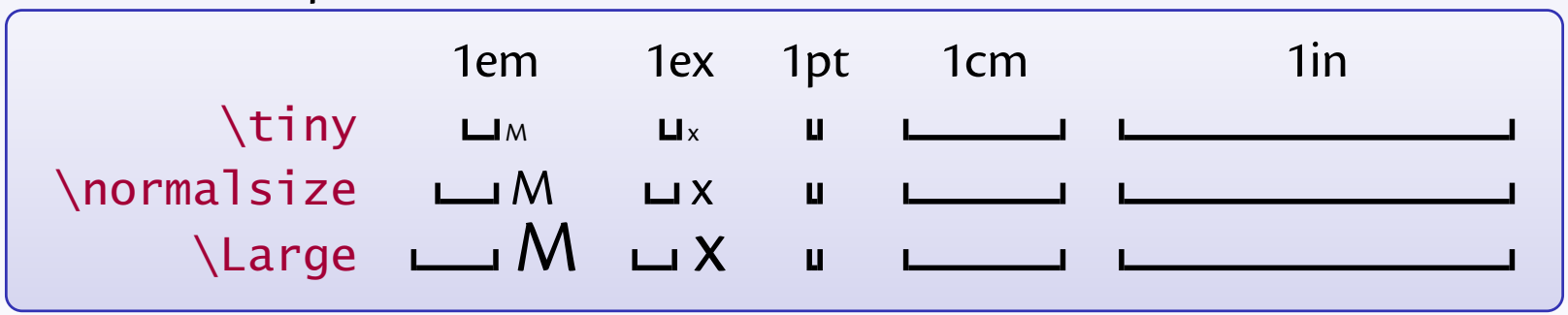
TEX peut manipuler des dimensions dans différentes unités :

`mm`	millimètre	`cm`	centimètre	`in`	*inch*	`pt`	point
`bp`	*big point*	`pc`	pica	`dd`	point Didot	`cc`	unité Cicéro
`sp`	*scaled point*	`ex`	hauteur de “x”	`em`	largeur de “M”	`mu`	*math unit*

Quelques facteurs de conversions :

1in = 2.54cm = 72.27pt = 72bp ; 1pc = 12pt ; 1157dd = 1238pt ; 1cc = 12dd

Les unités `em` et `ex` dépendent de la fonte de caractères courante :



En plus des dimensions fixes, TEX manipule des dimensions “élastiques”.
Leur syntaxe est de la forme `2cm plus 2mm minus 3mm`.
Ces “marges de manœuvre” sont utilisées pour ajuster les espaces horizontaux et verticaux.

Les dimensions sont des commandes qu’on ne peut pas invoquer seules :
`\parindent`, `\parsep`, `\baselineskip`, `\linewidth`, `\paperwidth`, `\textheight`, `\tabcolsep`, `\topmargin`, `\itemsep`, `\labelsep`…

Jouer avec les longueurs

LaTeX fournit des commandes pour gérer les dimensions.

`\newlength{\len}` définit la nouvelle commande de dimension `\len`.

Des opérations simples sont possibles :
`\setlength{\len}{⟨dim⟩}` assigne la valeur ⟨*dim*⟩ à `\len`,
`\addtolength{\len}{⟨dim⟩}` ajoute ⟨*dim*⟩ à `\len`.

Il est possible d'afficher une dimension avec la commande `\the` :

```
\newlength{\len} \setlength{\len}{3cm} \the\len,
\addtolength{\len}{1mm plus 2pt minus 3pc}\the\len
```
85.35826pt, 88.20352pt plus 2.0pt minus 36.0pt

En interne, TeX travaille en points, et donc les dimensions sont affichées dans cette unité.
On peut récupérer la valeur des longueurs dans la console de compilation ou dans le fichier .log en utilisant la commande `\typeout{\len=\the\len}` qui produit `\len=85.35826pt`.

Il existe une dimension élastique particulière : `\fill` peut aller de 0pt à ∞.
`\the\fill` ➞ 0.0pt plus 1.0fill
Cette "dimension" occupe l'espace disponible, comme un ressort en extension.
On peut augmenter la "dureté" du ressort.
La commande `\stretch{nbr}` vaut *nbr* fois `\fill` où *nbr* est un nombre entier :
`\the\stretch{4}` ➞ 0.0ptplus4fill

Faire de l'espace horizontalement

TEX ignore les espaces entre objets. Il existe des commandes pour créer de vrais espaces.

`\hspace{⟨dim⟩}` produit un espace horizontal de dimension ⟨*dim*⟩ :
`a b\hspace{1cm}c` ➟ a b c (1cm = ␣␣␣␣␣␣␣␣␣␣)
En début de ligne, l'espace est ignoré.
On peut le forcer avec la version étoilée : `\hspace*{⟨dim⟩}`

Des espacements horizontaux de taille prédéterminée sont fournis :
`\quad` est un espace de 1em : a b
`\enspace` est un espace de la moitié de `\quad` : a b
`\qquad` est un espace du double de `\quad` : a b
`\hfill` est un espace de dimension `\fill` : a b
(l'espace entre a et b est le plus grand possible, `\hfill` = `\hspace{\fill}`)
`\hrulefill` comme `\fill` avec un ligne horizontale : a_____b
`\dotfill` comme `\fill` avec une ligne pointillée : a . . . . . b
`\leftarrowfill` et `\rightarrowfill` : a←——————b——————→c

`\stretch{2}` est deux fois plus "puissant" que `\stretch{1}` (=`\fill`) :
`a\hspace{\stretch{1}}b\hspace{\stretch{1}}c` ➟ a b c
`a\hspace{\stretch{1}}b\hspace{\stretch{2}}c` ➟ a b c
On peut "doubler" la puissance d'un `\hrulefill` en en plaçant deux :
`a\hrulefill b\hrulefill\hrulefill c` ➟ a_____b_____c

Faire de l'espace verticalement

`\vspace{`⟨*dim*⟩`}` ajoute un espace vertical entre des paragraphes.
En début de page, cet espace vertical est supprimé :
il faut utiliser la version étoilée `\vspace*{`⟨*dim*⟩`}`
⚠ il ne faut pas utiliser cette commande dans un paragraphe, mais entre des paragraphes.

LaTeX définit 3 commandes d'espacement verticaux :
`\smallskip` est un espace vertical de dimension `\smallskipamount`, qui vaut environ un quart de `\baselineskip`
`\the\smallskipamount` ➞ 3.0pt plus 1.0pt minus 1.0pt

`\medskip` est un espace vertical de dimension `\medskipamount`, qui vaut environ la moitié de `\baselineskip`
`\the\medskipamount` ➞ 6.0pt plus 2.0pt minus 2.0pt

`\bigskip` est un espace vertical de dimension `\bigskipamount`, qui vaut environ `\baselineskip`
`\the\bigskipamount` ➞ 12.0pt plus 4.0pt minus 4.0pt

`\the\baselineskip` ➞ 12.0pt
⚠ la longueur `\baselineskip` n'est pas absolue, elle dépend de certains choix effectués dans le document, en particulier de la hauteur des caractères.

Les compteurs

Le langage de TEX contient la notion de compteurs.
Les compteurs ne sont pas des commandes. Ce sont des variables utilisées à de nombreux endroits : numérotation des pages, des formules, des items de listes…

Les compteurs par défaut définis par LATEX sont les suivants :

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

`mpfootnote` est le compteur des *footnotes* dans l'environnement `minipage`,
les compteurs `enumi` interviennent dans les listes.

`\newtheorem` définit des compteurs pour les théorèmes et autres structures sémantiques des mathématiques, d'autres *packages* peuvent fournir des compteurs…

À chaque compteur est associé une commande du type `\the`*ctr* qui affiche le compteur, où *ctr* est le nom du compteur : `\thepage` ➙ 32
Cette commande peut afficher plus que la valeur du compteur, par exemple `\theequation` peut rappeler le numéro de la section courante en affichant 2.4, où 2 est la valeur du compteur de section et 4 la valeur du compteur des équations.

Jouer avec les compteurs

LaTeX fournit des commandes pour gérer les compteurs.

`\newcounter{`*ctr*`}[`*masterctr*`]` définit le nouveau compteur *ctr*.
L'argument optionnel *masterctr* est un compteur "maître" existant : *ctr* est remis à 0 lorsque *masterctr* change.

Des opérations simples sont possibles :
`\setcounter{`*ctr*`}{`*val*`}` assigne la valeur *val* à *ctr*,
`\addtocounter{`*ctr*`}{`*val*`}` incrémente *ctr* de la valeur *val*,
`\stepcounter{`*ctr*`}` incrémente *ctr* d'une unité.

Le contenu d'un compteur est directement affichable sous différentes formes :

```
\arabic{-} \roman{-} \Roman{-} \alph{-} \Alph{-} \fnsymbol{-}
```

`\fnsymbol{-}` utilise une suite de 9 symboles ∗ † ‡ § ¶ ‖ ∗∗ †† ‡‡ en général utilisés pour la numérotation des notes de bas de page.

```
\newcounter{ctr} \setcounter{ctr}{4} \arabic{ctr}, \roman{ctr},
\Roman{ctr}, \alph{ctr}, \Alph{ctr}, \fnsymbol{ctr}.
```
4, iv, IV, d, D, §.

La commande `\value{`*ctr*`}` retourne la valeur du compteur *ctr* en tant que nombre
➡ utile dans les commandes qui demandent des nombres, par exemple comme second argument de `\setcounter`.

Redéfinir l'affichage d'un compteur

Les commandes `\the`*ctr* peuvent être rédéfinies par l'utilisateur.

`\renewcommand{\theequation}{\arabic{section}-\Alph{equation}}`
affiche les numéros d'équations sous la forme 2-D, où le compteur `section` vaut 2 et `equation` vaut 4.

`\renewcommand{\theequation}{\thesection-\Alph{equation}}`
réutilise l'affichage par défaut du compteur des sections.
➟ meilleure solution car respecte l'aspect d'un compteur déjà défini.

```
\renewcommand{\thesubsection}%
{{\bfseries\thesection\alph{subsection}}}
```
affiche les numéros des sous sections sous la forme **2c**.
(Noter l'usage d'un caractère `%` pour découper cette redéfinition en plusieurs lignes sans ajouter d'espace supplémentaire.)

Le *package* **amsmath** définit la commande `\numberwithin{-}{-}` qui assujettit le compteur donné dans le premier argument au compteur donné dans le second : `\numberwithin{subsection}{section}` remet le compteur des sous sections à 0 à chaque début de section et l'affichage est du type 2.3.
Les compteurs doivent exister avant d'invoquer cette commande.

Tour d'horizon de commandes de base

Où l'on se donne un peu de vocabulaire pour disposer à notre guise de la page blanche…

La justification

Par défaut, LaTeX justifie le texte dans les paragraphes : en jouant sur les espacements entre mots et les césures de mots, le texte est aligné sur les bords gauches et droits.
Les environnements `flushleft`, `flushright` et `center` désactivent les espacements "élastiques" entre les mots et les césures pour, respectivement, coller le texte sur le bord gauche, coller le texte sur le bord droit et centrer le texte.
Ces environnements acceptent des paragraphes et des ruptures de lignes avec `\\`.

```
\begin{center}
Actibus immensis\\
urbs fulget\\[2mm]
massiliensis
\end{center}
```

Actibus immensis
urbs fulget

massiliensis

Ces environnements ont des commandes (presque) équivalentes : `\raggedright`, `\raggedleft` et `\centering` (⚠ noms inversés !).

```
{\raggedleft Felix, qui potuit
rerum cognoscere causas\par}
```

Felix, qui potuit rerum
cognoscere causas

⚠ ces commandes ne fonctionnent que sur un paragraphe entier, d'où le `\par`.

Les listes

LaTeX définit trois types de listes : listes à puces, listes numérotées et listes à mots clés.

```
\begin{itemize}
\item In vino veritas
\item Natura abhorret a vacuo
\end{itemize}
```

- In vino veritas
- Natura abhorret a vacuo

```
\begin{enumerate}
\item Qui rogat, non errat
\item Qui scribit, bis legit
\end{enumerate}
```

1. Qui rogat, non errat
2. Qui scribit, bis legit

```
\begin{description}
\item[Cicéron :] Cuiusvis hominis est errare
\item[Horace :] Qui cupit aut metuit
liber non erit unquam
\end{description}
```

Cicéron : Cuiusvis hominis est errare

Horace : Qui cupit aut metuit liber non erit unquam

L'environnement de liste "brut", `list`, permet de définir d'autres types de listes.
Des environnements de type listes se cachent ailleurs : les environnements de bibliographie, d'index, de théorèmes…

Les listes (suite)

On peut emboîter jusqu'à 4 niveaux de listes :

```
\begin{enumerate}
  \item UN
    \begin{enumerate}
      \item un de UN
        \begin{enumerate}
          \item premier
          \item second
        \end{enumerate}
      \item deux de UN
    \end{enumerate}
  \item DEUX
\end{enumerate}
```

1. UN
    1. un de UN
        1. premier
        2. second
    2. deux de UN
2. DEUX

Les compteurs `enumi`, …, `enumiv` correspondent à ces emboîtements.
Les commandes `\labelenumi`… et `\labelitemi`… interviennent dans l'affichage des marques d'items, respectivement numérotés et non numérotés.
`\renewcommand{\labelitemiii}{+}` place un "+" comme marque d'item au niveau 3 de `itemize`, `\renewcommand{\labelenumii}{\theenumi-\theenumii}` redéfinit l'aspect des numéros au niveau 2 de `enumerate`.

Les tabulations

L'environnement `tabbing` définit un système de tabulations.

```
\begin{tabbing}
AAAAAA \= BBBBB \= CCCC \\
DD     \> EE    \> FF   \\
G      \> H     \> I    \\[3mm]
AAA    \= BB    \> CC   \\
à      \> é     \> ê
\end{tabbing}
```

AAAAAA BBBBB CCCC
DD EE FF
G H I

AAA BB CC
à é ê

`\=` place un taquet, `\>` positionne le texte qui suit sur le taquet suivant.

```
\begin{tabbing}
\hspace*{2cm}\=\hspace{1cm}\= \kill
a            \> b          \> c
\end{tabbing}
```

a b c

`\kill` n'affiche pas la ligne (➞ positionnement de taquets).
D'autres commandes de tabulations sont disponibles, `\'`, `\‘`, `\+`, `\-`, dont le comportement est trop long à expliquer ici.
⚠ Ne pas utiliser `\'e` et `\‘e` pour les lettres accentuées dans cet environnement.
Le *package* **Tabbing** règle ce problème.

Les tableaux

L'environnement `tabular` permet de composer des tableaux.

```
a \begin{tabular}[t]{||rc|l||}
ABC & DEF & GHI\\ \hline
J & K & L \\ \hline\hline
\multicolumn{2}{||c|}{MNO} & PQRS
\end{tabular} b
```

a

ABC	DEF	GHI
J	K	L
MNO		PQRS

b

Les colonnes peuvent être du type `r`, `c`, `l`, `p{`$\langle dim \rangle$`}` (paragraphe de largeur $\langle dim \rangle$).
L'argument optionnel de `tabular` gère l'alignement vertical du tableau.

On peut insérer du texte entre deux colonnes avec la syntaxe `r@{texte}l`

```
a \begin{tabular}[c]{r@{=}l}
d & 2 \\
c & 3
\end{tabular} b
```

a d=2 c=3 b

Différentes dimensions gèrent l'aspect des tableaux : `\tabcolsep` (séparation entre colonnes), `\arrayrulewidth` (épaisseur des traits), `\doublerulesep` (séparation entre les lignes doubles).
`\arraystretch` est un facteur multiplicatif de l'espacement entre les colonnes.

La gestion avancée des tableaux fera l'objet d'un autre cours.

Les flottants

Les éléments flottants sont des objets dont le placement exact dans le document est laissé à la sagacité de LaTeX. Les flottants sont conservés en mémoire qu'à rencontrer un endroit adéquat pour les placer (dans l'ordre d'arrivée).

➟ parfois cet endroit ne se rencontre qu'à la fin du document.

➟ un flottant peut bloquer le placement des flottants suivants.

Par défaut, LaTeX définit deux environnement flottants : `figure` et `table`.
`figure` s'utilise pour les graphiques, et `table` s'utilise pour les tableaux.

⚠ Pour insérer un graphique ou un tableau, on n'a pas besoin de ces environnements !

```
\begin{figure}[h]
  matériel de la figure
  \caption[Légende courte]{Un longue légende qui n'en finit pas}
  \label{fig-label}
\end{figure}
```

Ces deux environnements ont pour fonctions :

1. de placer leur contenu à un endroit du document où il y a suffisamment de place ;
2. de placer une légende numérotée renseignée par la commande `\caption[-]{-}`, de type `Figure` ou `Tableau` ;
3. de conserver les données (légende, page, numéro) dans un fichier auxiliaire pour construire une table des figures ou une table des tableaux.

Les flottants (suite)

L'option de ces environnements impose un certain comportement pour placer le flottant dans la page : `t` (haut de page), `b` (bas de page), `p` (page de flottants), `h` (ici si possible).
Ces options peuvent être combinées dans l'ordre de préférence : `[htp]`.
L'option `p` place le flottant sur une page *réservée* aux flottants.

Le *package* **`float`** ajoute l'option `H` qui place le flottant exactement à l'endroit défini.

Différents paramètres globaux (compliqués) permettent de gérer la façon dont les flottants sont placés dans le document.
`\topfraction` : proportion max. de la page réservée aux flottants en haut.
➟ `\renewcommand{\topfraction}{0.9}` permet de placer des flottants plus nombreux ou plus grands (0.7 par défaut).
`\bottomfraction` : proportion max. de la page pouvant contenir des flottants en bas.
`\textfraction` : proportion min. de la page contenant du texte.
➟ `\renewcommand{\textfraction}{0.1}` permet d'accepter plus de flottants (0.2 par défaut).
Voir aussi `\topnumber`, `\bottomnumber`, `\totalnumber`, `\floatpagefraction`.

La commande `\clearpage` (ou `\cleardoublepage` en mode `twoside`), interrompt la page en court, *place tous les flottants en attente*, et crée une nouvelle page.
➟ cette commande est appelée pour créer un nouveau chapitre par exemple.

La gestion avancée des flottants fera l'objet d'un autre cours.

La page de garde

Dans certaines classes, LaTeX compose une page de garde à partir de données spécifiques : `\title{-}` définit le titre, `\author{-}` renseigne les auteurs (séparés par la commande `\and`), `\date{-}` insère la date. Des commandes `\thanks{-}` associées aux auteurs produisent des informations en bas de page.
Après ces commandes, `\maketitle` compose le titre avec les éléments fournis.

Dans certaines classes, l'environnement `abstract` met en forme un résumé.

Les classes **`amsart`** et **`amsproc`** proposent des modifications : `\title[-]{-}` (titre court en option), `\translator{-}`, `\dedicatory{-}`, `\author[-]{-}` (nom court en option), `\contributor[-]{-}`, `\address{-}` (adresse postale), `\curraddr{-}` (adresse courante), `\email{-}`, `\urladdr{-}` (site Web), `\thanks{-}`, `\subjclass[-]{-}` (**Mathematics Subject Classification**), `\keywords{-}`…

La classe **`revtex`** propose les commandes suivantes : `\surname{-}` (pour l'indexation alphabétique), `\email[-]{-}`, `\homepage[-]{-}` (page Web), `\affiliation{-}` (adresse), `\altaffiliation[-]{-}` (adresse temporaire), `\collaboration{-}`, `\pacs{-}` (**Physics and Astronomy Classification Scheme**), `\keywords{-}`, `\preprint{-}`…

La classe **`svjour`** propose les commandes suivantes : `\subtitle{-}`, `\dedication{-}`, `\institute{-}`, `\email{-}`…

➙ on est loin d'une normalisation des métadonnées !

La bibliographie

L'environnement `thebibliography` démarre un nouveau chapitre ou une nouvelle section (selon la classe) et installe une liste pour présenter des références bibliographiques :

```
\begin{thebibliography}{00}
 \bibitem{ChamConn07} Ali~H. Chamseddine and Alain Connes.
  \newblock {Why the Standard Model}.
  \newblock {\em Journal of Geometry and Physics}, 58:38-47, 2008.

 \bibitem[Wei67]{Wein67a} Stephen Weinberg.
  \newblock A model of leptons.
  \newblock {\em Phys. Rev. Lett.}, 19(21):1264-1266, Nov 1967.
\end{thebibliography}
```

Les entrées sont libellées en interne par l'argument obligatoire de `\bibitem[-]{-}`.
Par défaut, les entrées ont un libellé apparent sous forme d'un numéro (dans l'ordre).
L'argument optionnel produit un libellé apparent personnalisé.
L'argument obligatoire de l'environnement `thebibliography` sert à fixer la largeur maximale des libellés apparents (seul le nombre de lettres compte !).
On peut citer un référence bibliographie avec `\cite[-]{-}` :
`\cite{ChamConn07}` ➡ [1], `\cite{Wein67a,ChamConn07}` ➡ [Wei67, 1],
`\cite[Thm~4]{ChamConn07}` ➡ [1, Thm 4]

La gestion avancée de la bibliographie (avec `bibtex`) fera l'objet d'un autre cours.

Commandes en vrac : gestion de la mise en page

Quelques commandes de mise en page :

- `\newpage` impose un saut de page.
  `\pagebreak[`*num*`]` suggère un saut de page.
  De *num* = 0 (on peut…) à *num* = 4 (on veut…).
  `\nopagebreak[`*num*`]` dissuade un saut de page.
  `\clearpage` traite les flottants en court et impose un saut de page.
  `\cleardoublepage` procède de même en commençant une nouvelle page à droite (mode `twoside`).
- `\enlargethispage{`⟨*dim*⟩`}` agrandit la page courante de la dimension spécifiée.
- `\indent` crée l'indentation en début de paragraphe (largeur `\parindent`).
  `\noindent` supprime l'indentation en début de paragraphe.
  La dimension `\parindent` correspond à cette indentation :
  ➡ `\setlength{\parindent}{0pt}` supprime toute indentation.
- `\newline` interrompt la ligne courante et en démarre un nouvelle.
  `\linebreak[`*num*`]` suggère une interruption de ligne.
  `\nolinebreak[`*num*`]` dissuade une interruption de ligne.
- `\footnote{-}` insère une note de bas de page.

Commandes en vrac : commandes de typographie

Quelques commandes de nature typographique :

- Les environnements `quote` et `quotation` servent à citer du texte : retraits du texte à droite et à gauche, `quotation` indente les paragraphes.
- L'environnement `verse` est destiné à la poésie.
- L'environnement `verbatim` sert à présenter du texte non interprété.
- Les guillemets (anglais) sont saisis avec la syntaxe ``` ``mot'' ``` ➟ "mot"
- Il y a trois sortes de tirets :
  `-` ➟ - (trait d'union), `--` ➟ – (tirets de listes), `---` ➟ — (début de dialogue)
- Les points de suspension sont obtenus pas `\dots` ou `\ldots` ➟ …
- Césures : TEX dispose d'une liste de césures.
  On peut localement l'aider en découpant un mot soi-même : `al\-gè\-bre`
  On peut le faire globalement : `\hyphenation{al-gè-bre vec-to-riel}`
  `\showhyphens{-}` affiche dans la console et le .log les suggestions de TEX :
  `\showhyphens{algèbre vectoriel}` ➟ `al-gèbre vec-to-riel`
- "Pavé noir" : `\rule[⟨dimh⟩]{⟨dimx⟩}{⟨dimy⟩}` créé un bloc plein de taille $\langle dimx \rangle \times \langle dimy \rangle$ relevé de la hauteur $\langle dimh \rangle$:
  `\rule{1cm}{3pt}` ➟ ▬,
  `x\rule[1ex]{1cm}{1pt}x` ➟ x‾‾‾‾‾x