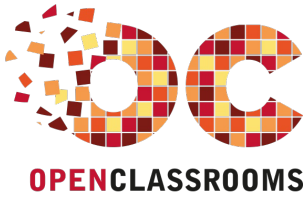
Comment utiliser la balise <math>

Par Les Validateurs
et Matthieu Schaller (Nanoc)



www.openclassrooms.com

Dernière mise à jour le 6/12/2011

Sommaire

Comment utiliser la balise <math>

Par  Matthieu Schaller (Nanoc) et  Les Validateurs

Mise à jour : 06/12/2011
Difficulté : Facile

La balise **<math>** du zCode (le langage utilisé sur ce site pour rédiger les messages de forum, les tutoriels ou les news) est une fonctionnalité plutôt méconnue de ce site, dont le but est de permettre l'insertion facile de formules mathématiques dans un texte. Insertion facile, car ces formules mathématiques sont tapées sous forme de texte directement dans la zForm (le formulaire permettant de taper du zCode, avec les smileys à gauche et les boutons en haut), plutôt que d'être insérées par l'auteur sous forme d'image difficilement modifiable.

Cependant, on reproche à cette balise sa certaine difficulté d'utilisation : en effet, pour permettre de réaliser des choses aussi bien basiques que très complexes avec **<math>**, la formule est décrite dans un langage compréhensible par certains logiciels, nommé le $\TeX$. Malgré sa syntaxe plutôt rebutante pour les débutants, $\TeX$ est un langage qui a fait ses preuves, étant notamment très utilisé dans la communauté scientifique. Le but de ce tutoriel est donc de vous apprendre à utiliser le langage $\TeX$ pour vous permettre d'utiliser au maximum la balise **<math>** lors de l'écriture de formules mathématiques.

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k + \int_a^x \frac{f^{(n+1)}(t)}{n!}(x-t)^n dt$$

Un exemple de formule mathématique rendue par $\TeX$.

Sommaire du tutoriel :

Écrivez vos premières formules !

Arrêtons ici le blabla et passons à un peu de pratique pour mieux comprendre comment $\TeX$ fonctionne et comment l'utiliser sur le Site du Zéro .

Comment insérer mes formules dans mes messages ?

Tout d'abord, comme pour tout ce qui est formatage de texte dans les messages de ce site, toute formule mathématique que vous voulez insérer doit être placée dans la balise **<math >** (donc entre **<math >** et **</math >** sans les espaces), comme ceci :

Code : Zcode

```
<math>2 + 2</math>
```

Résultat : $2 + 2$

Comme vous pouvez le remarquer, ces formules s'affichent directement au milieu du texte, sans effectuer de retour à la ligne avant ou après la balise (contrairement à la balise <?image> par exemple). Cela permet de les placer au milieu d'un raisonnement très facilement. Par exemple :

Code : Zcode

```
Tout le monde sait que <math>2 + 2 = 4</math>, c'est bien connu.
```

Résultat : Tout le monde sait que $2 + 2 = 4$, c'est bien connu.

TeX et ses commandes

Comme vous avez pu vous en rendre compte dans mes deux précédents exemples, $\TeX$ permet d'écrire très facilement des formules simples, en utilisant un format très intuitif (« 2 + 2 » génère par exemple le résultat attendu, $2 + 2$). Cependant, cette forme d'écriture très basique ne permet pas d'écrire des choses plus complexes, comme une multiplication :

Code : Zcode

```
<math>2 * 2</math>
```

Résultat : $2 * 2$

Comme vous pouvez vous en apercevoir, $\TeX$ remplace réellement l'étoile * par une étoile dans la formule alors qu'on voudrait l'utiliser comme une multiplication. On ne peut pas non plus utiliser **x** pour représenter la multiplication, car cette lettre est souvent utilisée pour représenter une variable. Ainsi, pour s'affranchir de ces limites de caractères utilisables, $\TeX$ permet d'utiliser toutes les fonctionnalités avancées sous forme de commandes.

Une commande, c'est tout simplement quelque chose qui sera traduit par $\TeX$ en quelque chose de différent de ce que vous avez tapé. En voilà un exemple, qui résout notre problème de la multiplication :

Code : Zcode

```
<math>2 \times 2</math>
```

Résultat : 2×2

Les commandes $\TeX$ commencent **toutes** (sans exception) par un *backslash* (la barre oblique de gauche à droite, \, accessible via AltGr+8 sur un clavier azerty). Il en existe un très grand nombre, et comme vous vous en doutez probablement, je vais tenter d'en détailler le maximum dans ce tutoriel pour vous permettre de faire des formules évoluées rapidement  .

Une question se pose maintenant : comment faire si vous voulez insérer un caractère *backslash* dans une formule $\TeX$? C'est simple : $\TeX$ vous fournit une commande permettant d'insérer ce caractère, qui est nommée **`\backslash`**  .

Nous avons résolu le problème de la multiplication, reste maintenant la division : comment réaliser une jolie fraction, avec le numérateur en haut, un trait, puis le dénominateur en bas ? Je suppose que vous avez maintenant deviné qu'il faudra également utiliser une commande. Cependant, vous ne savez pas comment spécifier à la commande appropriée que mettre en haut de la fraction et que mettre en bas. Nous allons maintenant nous intéresser au fonctionnement des commandes prenant des arguments pour réaliser ce rôle.

Argumentez !

Un argument, c'est un bout de formule que l'on donne à une commande pour qu'elle réalise un traitement dessus. Dans notre exemple, la commande réalisant une fraction prendra deux arguments : la formule à afficher au numérateur, et celle à afficher au dénominateur. Voici un exemple de formule affichant la fraction « 2 / 3 » :

Code : Zcode

```
<math>\frac{2}{3}</math>
```

Résultat : $\frac{2}{3}$

Vous l'avez probablement remarqué, les arguments se mettent entre accolades ({ et }, AltGr+4 et AltGr+= sur un clavier azerty), autant de fois qu'il faut d'arguments (ainsi, si la commande demande deux arguments, vous devez mettre deux paires d'accolades sans rien les séparant, comme dans l'exemple). Entre ces accolades, vous pouvez mettre ce que vous voulez, et bien entendu même du $\TeX$:

Code : Zcode

```
<math>\frac{2 \times 4}{8 \times 3}</math>
```

Résultat : $\frac{2 \times 4}{8 \times 3}$

Rien ne vous empêche d'ailleurs de mettre des fractions sur des fractions, même si le résultat n'est pas très lisible :

Code : Zcode

```
<math>\frac{\frac{x}{2}}{x \times \frac{y}{8}}</math>
```

Résultat : $\frac{\frac{x}{2}}{x \times \frac{y}{8}}$

Comme vous pouvez le voir, mettre des commandes à arguments comme arguments d'autres commandes ne pose strictement aucun problème, tant que vos arguments sont bien contenus dans une paire d'accolades.

Cependant, vous ne savez toujours pas tout sur les arguments ! Il existe une catégorie un peu à part d'arguments qui sont les arguments *optionnels* : cela signifie que vous n'êtes pas obligés de les préciser à la commande. Un exemple très simple est la fonction racine, accessible avec la commande suivante :

Code : Zcode

```
<math>\sqrt{100} = 10</math>
```

Résultat : $\sqrt{100} = 10$

Nous avons ici fait une racine carrée. Imaginons que dans un moment de folie mathématique nous voulions écrire une racine cubique : pour cela, il faut savoir que la commande **\sqrt** prend un argument optionnel permettant de donner le degré de la racine : 2 pour une racine carrée, 3 pour une racine cubique, etc. Voyons comment utiliser cet argument optionnel :

Code : Zcode

```
<math>\sqrt[3]{1000} = 10</math>
```

Résultat : $\sqrt[3]{1000} = 10$

Les crochets ([et]) servent à spécifier la valeur de cet argument optionnel. Il doit être placé avant tout argument obligatoire, et pas après : dans le cas contraire, $\TeX$ ne le traitera pas.

Et les notations dans tout ça ?

Vous avez vu dans un exemple plus haut qu'il ne pose aucun problème de mettre des variables comme « x » ou « y » dans votre équation. Cependant, en mathématiques, on est souvent amené à utiliser des choses comme des lettres grecques, des indices, des exposants, des ensembles, et plein de trucs du genre. Bien entendu, TEX vous permet cela de manière très aisée :

Code : Zcode

```
<math>\alpha + \beta + \gamma = \delta</math>
```

Résultat : $\alpha + \beta + \gamma = \delta$

Pour faire ces mêmes lettres en majuscule, mettez la première lettre de la commande en majuscule, comme ceci :

Code : Zcode

```
<math>\delta \times \Delta + \psi \times \Psi = \pi \times \Pi</math>
```

Résultat : $\delta \times \Delta + \psi \times \Psi = \pi \times \Pi$

Attention, toutes les lettres grecques n'existent pas en majuscule : si vous voulez par exemple obtenir un α majuscule, vous obtiendrez un joli , signe d'une erreur (commande inconnue).

Pour la notation ensembliste, on utilise la commande **mathbb** qui prend en argument le nom de l'ensemble à afficher :

Code : Zcode

```
<math>\mathbb{N} \times \mathbb{Q} \times \mathbb{R}</math>
```

Résultat : $\mathbb{N} \times \mathbb{Q} \times \mathbb{R}$

Pour les indices et les exposants, rien de plus simple : mettez simplement un caractère de soulignement _ (aussi appellé *underscore*, sur la touche 8 du clavier azerty) après un symbole, puis faites-le suivre par l'indice du symbole. Pour les exposants, remplacez simplement l'underscore par le chapeau ^ :

Code : Zcode

```
<math>i_0 \times u^2 = i_4^9</math>
```

Résultat : $i_0 \times u^2 = i_4^9$

Comme vous pouvez le constater, vous avez tout à fait le droit de mettre un indice suivi d'un exposant. Vous pouvez également mettre des expressions plus compliquées en exposant ou en indice, simplement en mettant des accolades autour de l'expression à mettre en indice ou exposant :

Code : Zcode

```
<math>i^{\frac{1}{2}}</math>
```

Résultat : $i^{\frac{1}{2}}$

Il est aussi possible de placer des exposants et des indices avant l'expression, par exemple pour des symboles chimiques. Notez également la commande **\mathrm** permettant de mettre son argument dans une police non oblique (comparez : U U).

Code : Zcode

```
<math>^{235}_{92}\mathrm{U}</math>
```

Résultat : $\mathrm{^{235}_{92}U}$

Vous connaissez maintenant le minimum vital pour taper des formules simples avec $\TeX$. Pour la petite anecdote, sachez que $\TeX$ propose une commande qui affiche son propre logo : il s'agit de la commande `\TeX`, que j'utilise depuis à peu près le début de ce tutoriel .

Passons maintenant à des choses plus sérieuses, avec des commandes plus avancées et à vocation plus spécifiques : comment créer un vecteur, afficher des opérateurs logiques, et plein d'autres sujets intéressants mais facultatifs selon l'utilisation que vous aurez de $\TeX$.

À partir d'ici, je ne mettrai plus les balises `<math>` dans les exemples de ce tutoriel : à vous de les rajouter selon vos besoins .

Symboles et notations plus évoluées

Dans cette partie, je vais vous donner un peu plus de symboles permettant de réaliser des formules complexes (toute blague disant que mes formules complexes sont imaginaires ne comptera pas, vous êtes prévenus). Je ne vais pas tous les expliquer, mais chaque catégorie sera illustrée par un ou deux exemples montrant une utilisation de ces symboles. Disons que cette partie a plus vocation de « mémo » que de vrai guide, vu que tout le nécessaire pour la comprendre a été donné plus haut .

Opérateurs de relation

Symbole	Commande	Description
$<$	`<`	Strictement inférieur à
$>$	`>`	Strictement supérieur à
$\leq$	`\leq`	Inférieur ou égal à
$\geq$	`\geq`	Supérieur ou égal à
$\subset$	`\subset`	Strictement inclus dans
$\supset$	`\supset`	Strictement compris dans
$\subseteq$	`\subseteq`	Inclus ou égal à
$\supseteq$	`\supseteq`	Compris ou égal à
$\equiv$	`\equiv`	Équivalent à
$\sim$	`\sim`	Tilde (relations d'équivalence)
$=$	`=`	Égal à
$\neq$	`\neq`	Différent de
$\perp$	`\perp`	Perpendiculaire
$\parallel$	`\parallel`	Parallèle

Exemples :

$\mathbb{N} \subset \mathbb{R}$	`\mathbb{N} \subset \mathbb{R}`
$3 \neq 4$	`3 \neq 4`
$(Ox) \perp (Oy)$	`(Ox) \perp (Oy)`

Flèches

Symbole	Commande	Description
$\leftarrow$	`\leftarrow`	Flèche simple vers la gauche
$\rightarrow$	`\rightarrow`	Flèche simple vers la droite
$\leftrightarrow$	`\leftrightarrow`	Flèche simple pointant des deux côtés
$\Leftarrow$	`\Leftarrow`	Implication réciproque
$\Rightarrow$	`\Rightarrow`	Implication directe
$\Leftrightarrow$	`\Leftrightarrow`	Equivalence
$\longleftarrow$	`\longleftarrow`	Flèche simple vers la gauche (longue)
$\longrightarrow$	`\longrightarrow`	Flèche simple vers la droite (longue)
$\longleftrightarrow$	`\longleftrightarrow`	Flèche simple pointant des deux côtés (longue)
$\Longleftarrow$	`\Longleftarrow`	Implication réciproque (flèche longue)
$\Longrightarrow$	`\Longrightarrow`	Implication directe(flèche longue)
$\Longleftrightarrow$	`\Longleftrightarrow`	Equivalence (flèche longue)

Exemples :

$a = b \Longleftrightarrow b = a$	`a = b \Longleftrightarrow b = a`
$f(x) = x^2 \Longrightarrow f'(x) = 2x$	`f(x) = x^2 \Longrightarrow f'(x) = 2x`
$\|x\| + 2 = 3 \Longleftarrow x = 1$	`|x| + 2 = 3 \Longleftarrow x = 1`

Chapeaux et symboles à appliquer aux variables

Symbole	Commande	Description
$\bar{A}$	`\bar{A}`	Barre "au dessus"
$\vec{v}$	`\vec{v}`	Vecteur
$\dot{x}$	`\dot{x}`	Première dérivée temporelle en physique
$\ddot{x}$	`\ddot{x}`	Seconde dérivée temporelle en physique
$\widehat{ABC}$	`\widehat{ABC}`	Angle
$\overbrace{abc}$	`\overbrace{abc}`	Annotation, via une accolade horizontale par dessus la formule
$\underbrace{abc}$	`\underbrace{abc}`	Annotation, via une accolade horizontale sous la formule

Quantifieurs et autres symboles utiles

Symbole	Commande	Description
$\forall$	`\forall`	Pour tout
$\in$	`\in`	Appartient à
$\notin$	`\notin`	N'appartient pas à
$\exists$	`\exists`	Il existe
$\approx$	`\approx`	Environ égal à
$\simeq$	`\simeq`	Similaire à
$\pm$	`\pm`	Plus ou moins

Exemples :

$\forall \epsilon, \exists r, \forall x$	`\forall \epsilon,\ \exists r,\ \forall x`
$0.5 \pm 10^{-6} \approx 0.5$	`0.5 \pm 10^{-6} \approx 0.5`

Opérateurs ensemblistes

Symbole	Commande	Description
$\cup$	`\cup`	Union
$\cap$	`\cap`	Intersection
$\setminus$	`\setminus`	Privé de
$\oplus$	`\oplus`	Somme directe

Exemples :

$\mathbb{N}^{-*} \cup \mathbb{N}^{+*} = \mathbb{N} \setminus \{0\}$	`\mathbb{N}^{\tiny{-*}}\ \cup\ \mathbb{N}^{\tiny{+*}} = \mathbb{N} \setminus \{0\}`
$\mathbb{A} = \mathbb{B} \oplus \mathbb{C}$	`\mathbb{A} = \mathbb{B} \oplus \mathbb{C}`

Sigma, pi et limites

J'ai nommé ces trois symboles en particulier pour une raison simple : ils sont les plus connus et les plus utilisés d'une liste de 10 « grands » symboles qui s'utilisent de manière différente des autres : en effet, si on prend comme exemple la somme $\sum$, elle prend deux « paramètres » : le nombre de départ et de fin de la somme. Or, pour une raison ou pour une autre, ces paramètres ne sont pas donnés comme pour les autres commandes $\TeX$: on utilise pour les placer la notation exposant ou indice. Un exemple pour y voir plus clair ? 

Code : TeX

```
\sum_{i = 0}^5 i = 0 + 1 + \ldots + 5 = 15
```

Résultat : $\sum_{i=0}^{5} i = 0 + 1 + \ldots + 5 = 15$

Ainsi, ce que nous avions placé en indice du symbole somme se retrouve situé en-dessous, et ce qui a été placé en exposant se retrouve au-dessus, pratique ! Les autres symboles se comportant ainsi sont les suivants :

- Le produit $\prod$ (`\prod`)
- Le co-produit $\coprod$ (`\coprod`)
- L'intégrale $\int$ (`\int`)
- La limite $\lim$ (`\lim`)
- L'union d'ensembles $\bigcup$ (`\bigcup`)
- L'intersection d'ensembles $\bigcap$ (`\bigcap`)

Un autre exemple pour cette partie, cette fois d'une intégrale :

Code : TeX

```
\int_0^x t^2 \mathrm{d}t = \frac{x^3}{3}
```

Résultat : $\int_0^x t^2 \mathrm{d}t = \frac{x^3}{3}$

Pour finir, lors de l'écriture de limites, il est plus que fréquent d'utiliser le symbole « infini ». Pour cela, la commande à utiliser est `\infty` (et pas `\inf` ou `\infinity` , ça ne marchera pas !). Par exemple :

Code : TeX

```
\lim_{x \to +\infty} \frac{1}{x} = 0
```

Résultat : $\lim_{x \to +\infty} \frac{1}{x} = 0$

Le cas des fonctions

Il peut arriver de devoir utiliser des fonctions dans vos formules. Les exemples les plus courants sont les très connues fonctions cosinus et sinus. La manière intuitive de les écrire, $tan(x)$ (`tan(x)`) est peu esthétique : en effet, $\TeX$ considère dans ce cas que vous voulez écrire $t \times a \times n(x)$ sous une forme plus compressée (comme on pourrait par exemple écrire $4\omega\, Rt$). Pour ces raisons, les fonctions trigonométriques possèdent toutes une commande qui leur est associée et permettant une écriture plus propre de ces dernières : par exemple, dans $tan(x) \approx \tan(x)$ (`tan(x) \approx \tan(x)`), la deuxième tangente est plus jolie que la première, car elle a été écrite avec la commande appropriée, `\tan`.

Bien entendu, cette astuce est disponible pour toutes les autres fonctions trigonométriques et quelques autres fonctions comme les logarithmes ou les exponentielles :

- $cos(x) \approx \cos(x)$: `cos(x) \approx \cos(x)`
- $sin(x) \approx \sin(x)$: `sin(x) \approx \sin(x)`
- $tan(x) \approx \tan(x)$: `tan(x) \approx \tan(x)`
- $cosh(x) \approx \cosh(x)$: `cosh(x) \approx \cosh(x)`
- $sinh(x) \approx \sinh(x)$: `sinh(x) \approx \sinh(x)`
- $tanh(x) \approx \tanh(x)$: `tanh(x) \approx \tanh(x)`
- $arccos(x) \approx \arccos(x)$: `arccos(x) \approx \arccos(x)`
- $arcsin(x) \approx \arcsin(x)$: `arcsin(x) \approx \arcsin(x)`
- $arctan(x) \approx \arctan(x)$: `arctan(x) \approx \arctan(x)`

- $ln(x) \approx \ln(x)$: `ln(x) \approx \ln(x)`
- $log(x) \approx \log(x)$: `log(x) \approx \log(x)`
- $exp(x) \approx \exp(x)$: `exp(x) \approx \exp(x)`

Cette liste est non exhaustive : je ne promets pas que les fonctions n'apparaissant pas dedans ne sont pas disponibles sous la forme d'une commande TEX. De ce fait, essayez toujours de faire précéder les noms des fonctions par un backslash pour ainsi tester l'existence d'une commande.

Encore une fois, rien de compliqué sur ce point  . Passons maintenant à quelque chose d'un peu plus évolué avec l'écriture de matrices et de vecteurs.

Écriture de matrices

Les matrices font probablement partie des choses les plus avancées que vous aurez à écrire avec TEX. Pour faire comprendre à TEX que ce que vous allez écrire doit être formaté sous la forme d'une matrice, il va falloir utiliser un outil très pratique appelé environnement. Voyons maintenant en quoi cela consiste  .

Les environnements, qu'est-ce que c'est ?

Un environnement est un morceau de votre formule contenu entre deux commandes particulières : **\begin** et **\end**. Ces deux commandes prennent un argument qui est le nom de l'environnement à ouvrir ou à fermer. L'environnement va gérer l'affichage de son contenu comme il le veut, selon des règles qui lui sont propres. Voyons pour notre exemple l'environnement qui nous permet d'écrire des matrices, nommé **matrix**.

Comme vous l'avez sûrement compris, pour indiquer à TEX que vous souhaitez afficher une matrice, il faut imbriquer le contenu de votre matrice entre les deux commandes **\begin**{matrix} et **\end**{matrix}. Reste maintenant à voir comment écrire le contenu de votre matrice et comment dire à TEX où placer les expressions.

Votre première matrice en TeX

L'écriture du contenu est elle aussi simplifiée à l'extrême : chaque colonne doit être séparée par un caractère **ampersand** (**&**, le symbole sur la touche du 1 sur un clavier azerty français), alors que chaque ligne doit être séparée par deux **backslashes** (\\, avec le même caractère que vous utilisez au début d'une commande). Ainsi, si on veut faire une matrice de 2x2 contenant a, b, c, et d, le code ressemble à ceci :

Code : TeX

```
\begin{matrix}
a & b \\
c & d
\end{matrix}
```

Résultat : $\begin{matrix} a & b \\ c & d \end{matrix}$

Première remarque, vous pouvez bien entendu mettre du code sur plusieurs lignes dans le contenu de la balise **<math>** : TEX ignorera silencieusement les sauts de lignes. Deuxième remarque, le rendu est moche, car rien n'est mis autour de la matrice, alors qu'on voudrait dans la plupart des cas avoir le contenu entre parenthèses. Pour cela, on préférera utiliser l'environnement **pmatrix** qui s'utilise exactement de la même manière que l'environnement **matrix** :

Code : TeX

```
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
```

Résultat : $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

C'est tout de suite plus agréable. Bien entendu, j'ai ici utilisé des expressions simples, mais vous pouvez placer des expressions beaucoup plus compliquées comme contenu de vos matrices :

Code : TeX

```
\begin{pmatrix}
R \times \cos(\theta) & R \times \sin(\theta) \\
\arctan(\phi) + e^{4i + 2} & \sqrt[4]{\ln(\cos(\frac{3 \pi}{16}))}
\end{pmatrix}
```

Résultat : $\begin{pmatrix} R \times \cos(\theta) & R \times \sin(\theta) \\ \arctan(\phi) + e^{4i+2} & \sqrt[4]{\ln(\cos(\frac{3\pi}{16}))} \end{pmatrix}$

À partir de là, vous pouvez également écrire des vecteurs sous la forme de matrice à une colonne :

Code : TeX

```
\begin{pmatrix}
x \\
y \\
z
\end{pmatrix}_{(\vec{e_x}, \vec{e_y}, \vec{e_z})}
```

Résultat : $\begin{pmatrix} x \\ y \\ z \end{pmatrix}_{(\vec{e_x}, \vec{e_y}, \vec{e_z})}$

Attention, ne confondez pas un vecteur à deux dimensions $\begin{pmatrix} a \\ b \end{pmatrix}$ avec l'écriture des combinaisons « p parmi n » $\begin{pmatrix} n \\ p \end{pmatrix}$ (`n \choose p`) : en effet, même si ces deux écritures se ressemblent beaucoup, elles n'ont pas du tout la même signification. Veillez donc à ne pas les interchanger  .

Astuces diverses

Pour finir ce tutoriel, je vais dans cette sous-partie vous introduire quelques astuces utiles dans l'écriture de formules mathématiques plus complexes. Commençons  .

Des grands délimiteurs : parenthèses, crochets, accolades

Vous avez surement remarqué que lorsque vous entourez par exemple une fraction de parenthèses, les parenthèses ont une taille ne correspondant pas du tout à celle de la fraction : par exemple, $(\frac{1}{2}) \times (\frac{5}{6})$. Pour cela, deux commandes existent : `\left` et `\right` : elles permettent de donner la taille adéquate à des délimiteurs comme les parenthèses, les crochets ou les accolades. Par exemple :

Code : TeX

```
\left( \frac{1}{2} \right)
```

Résultat : $\left(\frac{1}{2} \right)$

Vous pouvez bien entendu imbriquer ces différents délimiteurs :

Code : TeX

```
\left[ \left( \frac{1}{2} \right) \left( \frac{6}{5} \right) \right]
```

Résultat : $\left[\left(\frac{1}{2}\right) \times \left(\frac{6}{5}\right)\right]$

Devenez le maître de l'espacement

Il est parfois utile de pouvoir séparer plusieurs parties de vos formules. Pour cela, utilisez les commandes suivantes (liste honteusement reprise de la Wikipédia) :

- `a \qquad b` donne $a \qquad b$
- `a \quad b` donne $a \quad b$
- `a\ b` donne $a\ b$
- `a\;b` donne $a\;b$
- `a\,b` donne $a\,b$
- `ab` donne ab
- `a\!b` donne $a\!b$

Vous pouvez avec ces commandes écrire des matrices de manière plus lisibles, en y insérant des espacements plus grands que ceux par défaut :

Code : TeX

```
\begin{pmatrix}
R \times \cos(\theta) & R \times \sin(\theta) \\
\arctan(\phi) + e^{4i + 2} & \sqrt[4]{\ln(\cos(\frac{3 \pi}{16}))}
\end{pmatrix}
\quad \times \quad
\begin{pmatrix}
R \times \cos(\theta) & R \times \sin(\theta) \\
\arctan(\phi) + e^{4i + 2} & \qquad \qquad \sqrt[4]{\ln(\cos(\frac{3
\pi}{16}))}
\end{pmatrix}
```

Résultat :

$$\begin{pmatrix} R \times \cos(\theta) & R \times \sin(\theta) \\ \arctan(\phi) + e^{4i+2} & \sqrt[4]{\ln(\cos(\frac{3\pi}{16}))} \end{pmatrix} \times \begin{pmatrix} R \times \cos(\theta) & R \times \sin(\theta) \\ \arctan(\phi) + e^{4i+2} & \sqrt[4]{\ln(\cos(\frac{3\pi}{16}))} \end{pmatrix}$$

Du texte dans vos formules !

Il est parfois utile de pouvoir mettre du texte dans le contenu de vos formules. Pour cela, utilisez la commande **\mbox** avec en argument le texte que vous souhaitez insérer. Par exemple, `x^2 = 3 \Longleftrightarrow x = \sqrt{3} \quad \mbox{ou} \quad x = -\sqrt{3}` donne $x^2 = 3 \Longleftrightarrow x = \sqrt{3} \quad \mbox{ou} \quad x = -\sqrt{3}$. La commande **\text** existe elle aussi et permet de réaliser la même chose.

Merci à Aravis pour l'idée de cette astuce.

Écrivez au-dessus et en-dessous des symboles

De la même manière que vous pouvez indiquer des expressions en indice ou en exposant d'autres expressions, vous pouvez également écrire au-dessus et en-dessous des symboles communs comme les flèches, comme ceci :

Code : TeX

```
\arctan(x) \sim_{+\infty} \frac{\pi}{2}
```

Résultat : $\arctan(x) \sim_{+\infty} \frac{\pi}{2}$

Pour un résultat plus esthétique, il peut être agréable que le texte placé au-dessus ou en-dessous des symboles soit écrit dans une police plus petite. Pour cela, combinez cette notation avec la commande `\tiny` :

Code : TeX

```
\arctan(x) \sim_{\tiny +\infty} \frac{\pi}{2}
```

Résultat : $\arctan(x) \sim_{\tiny +\infty} \frac{\pi}{2}$

Merci à Aravis pour l'idée de cette astuce.

L'environnement array pour écrire des équations

Quand vous êtes amenés à écrire de grands calculs sur plusieurs lignes, il est important de savoir aligner les différentes parties des égalités dans le but d'améliorer la lisibilité. Pour cela, on utilise l'environnement array : il est exactement identique à l'environnement matrix que nous avons vu dans la sous-partie précédente, mais sous un nom différent  . Par exemple :

Code : TeX

```
\begin{array}{cccc}
& x^2 & = & 5 \\
\Longleftrightarrow & |x| & = & \sqrt{5}
\end{array}
```

Résultat :

$$\begin{array}{cccc} & x^2 & = & 5 \\ \Longleftrightarrow & |x| & = & \sqrt{5} \end{array}$$

Cet environnement vous permet également de spécifier l'alignement du texte dans les colonnes, pour par exemple aligner à gauche ou à droite plutôt que de centrer l'expression. Un petit exemple :

Code : TeX

```
\begin{array}{lcr}
\text{A gauche} & \text{Au centre} & \text{A droite} \\
x & x & x
\end{array}
```

Résultat :

$$\begin{array}{lcr} \text{A gauche} & \text{Au centre} & \text{A droite} \\ x & x & x \end{array}$$

Si vous avez d'autres idées d'astuces du même genre, n'hésitez pas à me les faire parvenir en commentaire de ce tutoriel, je ne manquerai pas de les y ajouter si elles présentent un réel intérêt pour le lecteur  .

Q.C.M.

Le premier QCM de ce cours vous est offert en libre accès.
Pour accéder aux suivants

Connectez-vous Inscrivez-vous

Quel est le langage utilisé pour écrire les formules mathématiques sur le Site du Zéro ?

- ◯ MathML
- ◯ La balise <math>

- ◯ TeX
- ◯ La réponse D

Quel caractère introduit-il une commande $\TeX$?

- ◯ L'ampersand
- ◯ Le slash
- ◯ Le backslash
- ◯ Le dollar

Laquelle de ces commandes permet d'écrire une racine cubique ?

- ◯ sqrt
- ◯ curt
- ◯ pow
- ◯ frac

Quel environnement ne permet pas d'écrire des matrices ?

- ◯ matrix
- ◯ array
- ◯ pmatrix
- ◯ grid

Correction !

Statistiques de réponses au QCM

Voilà qui achève votre apprentissage des bases du $\TeX$. Désormais, une formule comme celle donnée en introduction ne devrait plus avoir de secrets pour vous (au moins au niveau de l'écriture). Vous pouvez maintenant utiliser correctement et efficacement la balise `<math>` du zCode sans difficulté, pour présenter des formules dans vos messages ou vos tutoriels .

Cependant, ce tutoriel est non exhaustif : $\TeX$ est un sujet très vaste impossible à traiter complètement dans un tutoriel court : cela n'aurait d'ailleurs que peu d'intérêt vu l'abondance de documentation sur le sujet. On peut par exemple citer la documentation officielle de mimeTeX (l'outil utilisé par le Site du Zéro pour transformer le $\TeX$ en images) ou le guide disponible sur Wikipédia.

Ce tutoriel a été corrigé par les zCorrecteurs.