

Les \TeX nicités d'une affiche

Jonathan Godin

Février 2019

Contenu

- 1 Informations importantes
- 2 Quand utiliser \LaTeX
- 3 Conception et gabarit
- 4 Ajuster la mise en page

Introduction

- Créer une affiche avec *Beamer* est plus facile qu'on le croit
- Il existe plusieurs gabarits (*templates*)
- On peut personnaliser et ajuster la mise en page avec certaines *commandes primitives*

Informations importantes

- Le temps d'impression de **24 à 48 heures**
 - ▶ Votre affiche doit être **prête** à imprimer **lundi le 4 mars**
- Il n'y a pas de format standard en tant que tel, mais
 - ▶ 36" × 46" (Gros) à peu près le format A0
 - ▶ 24" × 36" (moyen) à peu près A1
 - ▶ 11" × 17" (très petit)
- 52" est la largeur maximale
- Pas de longueur maximale

Limitation de \LaTeX /Beamer

Un peu de théorie

Pour \TeX , tout est un **rectangle**! Les caractères sont des rectangles que l'on colle pour former des lignes, qui sont des rectangles que l'on colle pour former des paragraphes, qui sont des rectangles que l'on colle pour former une page.

Exemple

- Quand on voit

Weniges, aber Reifes

- \TeX voit



Voir Chp. 11 du \TeX book “Boxes”

(Ne pas se mêler avec le chp. 21 “Making Boxes” !)

Limitation de L^AT_EX/Beamer

- Il faut penser en terme de rectangles
- Si les rectangles ne sont pas adéquats pour le concept
 - ▶ il vaut mieux ne pas utiliser L^AT_EX/Beamer
- Petite exception aux rectangles : `\parshape`

Conception et gabarit

- Ayez en tête le concept de votre affiche, car :

Surprise !

Un gabarit ne fait pas de miracle

- Un gabarit s'occupe de :
 - ▶ la taille de l'affiche (format)
 - ▶ la taille et le style des polices de caractères (modifiables)
 - ▶ certains éléments de présentation (le thème du Beamer)

block

Un bloc

alertblock

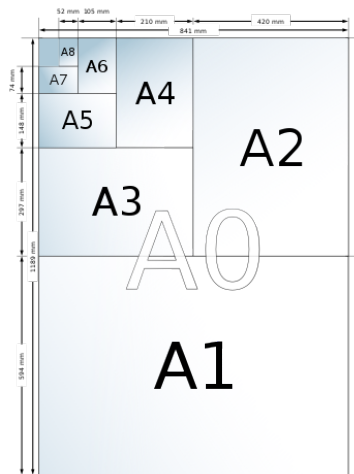
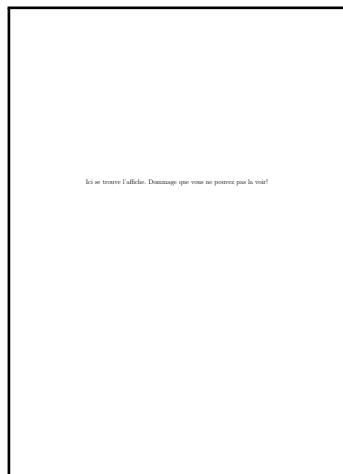
Une alert :O

exampleblock

Un exemple ?

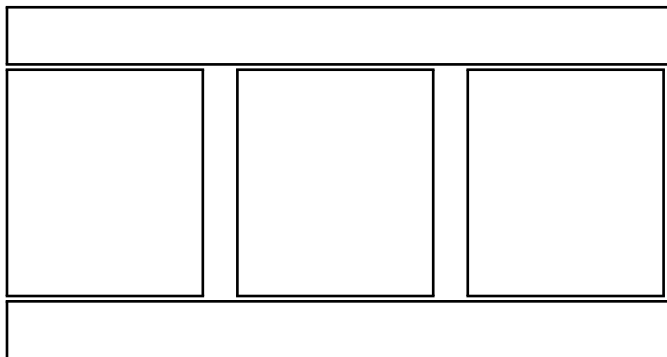
Premier exemple de gabarit

```
\usepackage[orientation=portrait,size=a0,  
scale=1.4,debug]{beamerposter}
```



(image de wikipédia)

Utiliser les colonnes




Utiliser les colonnes

<Entête et titre>

```
\begin{columns}
  \begin{column}{.33\textwidth}
    :
  \end{column}
  \begin{column}{.33\textwidth}
    :
  \end{column}
  \begin{column}{.33\textwidth}
    :
  \end{column}
\end{columns}
```

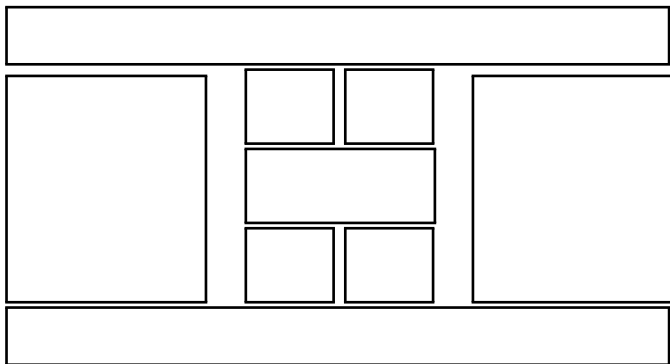
<Bas de page et conclusion>

Utiliser les colonnes

```
\begin{columns}
  \begin{column}{.33\textwidth}
    
  \end{column}{.33\textwidth}
  \vdots
\end{columns}
```

← en utilisant
d'autres
colonnes

Utiliser les colonnes



- Moral : Le gabarit et le thème gère l'apparence, utilisez ensuite les colonnes pour obtenir le format voulu

Deuxième exemple de gabarit

- En fait, le premier exemple est général !

Le package beamerposter

- En fait, le premier exemple est général !
- Toutes les affiches ne sont qu'une combinaison astucieuse de colonnes
- Utiliser

```
\usepackage[orientation=<portrait|landscape>,  
  size=<a0b|a0|a1|a2|a3|a4|custom>,  
  width=<dimen>, height=<dimen>, % Si size=custom  
  scale=<float>, % facteur de dilatation  
  % des polices de caractère  
  debug] % pour utiliser le package fp  
{beamerposter}
```

Ajuster la mise en page

- But de \LaTeX : *Contrôler* la mise en page pour que l'auteur se concentre sur le contenu
- Sur une affiche, il est parfois *nécessaire* de gérer la mise en page soi-même
- Certaines commandes primitives de \TeX contournent l'emprise de \LaTeX sur la mise en page

Primitives

<code>\vskip</code>	<code>\hskip</code>	<code>\par</code>	<code>\input</code>
<code>\write</code>	<code>\immediate</code>	<code>\hbox</code>	<code>\vbox</code>
<code>\vtop</code>	<code>\vcenter</code>	<code>\leaders</code>	<code>\vrule</code>
<code>\hrule</code>	<code>\string</code>	<code>\the</code>	<code>\show</code>
<code>\meaning</code>	<code>\showthe</code>	<code>\font</code>	<code>\special</code>
<code>\kern</code>	<code>\raise</code>	<code>\lower</code>	<code>\catcode</code>
<code>\edef</code>	<code>\xdef</code>	<code>\global</code>	<code>\advance</code>
<code>\multiply</code>	<code>\expandafter</code>	<code>\noexpand</code>	<code>\let</code>
<code>\halign</code>	<code>\valign</code>	<code>\vadjust</code>	<code>\setbox</code>
<code>\parshape</code>	<code>\leftskip</code>	<code>\rightskip</code>	<code>\accent</code>
<code>\over</code>	<code>\atop</code>	<code>\tolerance</code>	<code>\ignorespaces</code>

Primitives

<code>\vskip</code>	<code>\hskip</code>	<code>\par</code>	<code>\input</code>
<code>\write</code>	<code>\immediate</code>	<code>\hbox</code>	<code>\vbox</code>
<code>\vtop</code>	<code>\vcenter</code>	<code>\leaders</code>	<code>\vrule</code>
<code>\hrule</code>	<code>\string</code>	<code>\the</code>	<code>\show</code>
<code>\meaning</code>	<code>\showthe</code>	<code>\font</code>	<code>\special</code>
<code>\kern</code>	<code>\raise</code>	<code>\lower</code>	<code>\catcode</code>
<code>\edef</code>	<code>\xdef</code>	<code>\global</code>	<code>\advance</code>
<code>\multiply</code>	<code>\expandafter</code>	<code>\noexpand</code>	<code>\let</code>
<code>\halign</code>	<code>\valign</code>	<code>\vadjust</code>	<code>\setbox</code>
<code>\parshape</code>	<code>\leftskip</code>	<code>\rightskip</code>	<code>\accent</code>
<code>\over</code>	<code>\atop</code>	<code>\tolerance</code>	<code>\ignorespaces</code>

Primitives

<code>\vskip</code>	<code>\hskip</code>	<code>\par</code>	<code>\input</code>
<code>\write</code>	<code>\immediate</code>	<code>\hbox</code>	<code>\vbox</code>
<code>\vtop</code>	<code>\vcenter</code>	<code>\leaders</code>	<code>\vrule</code>
<code>\hrule</code>	<code>\string</code>	<code>\the</code>	<code>\show</code>
<code>\meaning</code>	<code>\showthe</code>	<code>\font</code>	<code>\special</code>
<code>\kern</code>	<code>\raise</code>	<code>\lower</code>	<code>\catcode</code>
<code>\edef</code>	<code>\xdef</code>	<code>\global</code>	<code>\advance</code>
<code>\multiply</code>	<code>\expandafter</code>	<code>\noexpand</code>	<code>\let</code>
<code>\halign</code>	<code>\valign</code>	<code>\vadjust</code>	<code>\setbox</code>
<code>\parshape</code>	<code>\leftskip</code>	<code>\rightskip</code>	<code>\accent</code>
<code>\over</code>	<code>\atop</code>	<code>\tolerance</code>	<code>\ignorespaces</code>

Aligner à l'aide d'espace infiniment élastique

Primitives :

- `\hfil`, `\hfill`,
`\hskip0pt plus 1filll`

- `\vfil`, `\vfill`,
`\vskip0pt plus 1filll`

- Par exemple
- il y a
- un `\vfill`
- entre chaque item de cette liste



Crénage entre les gros caractères (kerning)

- Les titres et sous-titres peuvent être très gros (60 à 80pt)

- P.ex., ce **mot** est en 80pt

- L'espacement est très mauvais
 - ▶ On utilise `\kern` pour le modifier

- Comme ceci : **mot**
 - ▶ `m\kern-7pt o\kern-5pt t`

Polices de caractères (X_YL^AT_EX)

- La primitive `\font` permet d'appeler une nouvelle fonte
- `\font` de X_YL^AT_EX est beaucoup plus puissant
 - ▶ Ex : Cette phrase est en Linux Libertine de taille 12pt.
 - ▶ `\font\liber="Linux Libertine 0" at 12pt`
`{\liber Cette phrase est en Linux Libertine de`
`taille 12pt.}`
- Elle permet d'utiliser n'importe quelle fonte
- `\font` est utile et rapide pour utiliser une fonte une fois
- Pour changer la police pour l'ensemble de l'affiche, il vaut mieux utiliser un package

`\strut`

- Un bloc est un peu plus profond que le caractère le plus profond.
 - ▶ Pour du gros texte, ce n'est pas toujours assez profond

Mot

- On utilise `\strut` pour insérer un caractère invisible plus profond : |
- ex. avec `\fbox` : `\fbox{a}` et `\fbox[\strut]{a}`

Mot

- Primitive : `\vrule width* height* depth*`
- `\vrule width 2cm height 3pt depth-2pt`

Problème avec `\centering` pour les titres

Le plus grand problème d'analyse est que le mot « analyse » n'ait pas d'abréviation appropriée.

Le plus grand problème d'analyse est que le mot « analyse » n'ait pas d'abréviation appropriée.

Remplacer `\centering` par `\raggedcenter`

On a obtenu

Le plus grand problème d'analyse est que le mot
« analyse » n'ait pas d'abréviation appropriée.

avec

`\raggedcenter`

```
macro :->\leftskip =0pt plus4em \rightskip =\leftskip  
\parfillskip =0pt \spaceskip =.3333em \xspaceskip =.5em  
\pretolerance =9999 \tolerance =9999 \parindent =0pt  
\hyphenpenalty =9999 \exhyphenpenalty =9999
```

- `\raggedcenter` n'est pas défini dans le format \LaTeX
 - ▶ `\def\raggedcenter{\leftskip = 0pt plus 4em ...}`
- `*\usepackage[newcommands]{ragged2e}`
 - ▶ Ceci redéfinit `\centering` de la bonne façon

\parshape

`\parshape=n i1 ℓ1 i2 ℓ2 ... in ℓn`

- n est le nombre de ligne
- ℓ_j la longueur de la ligne j
- i_j l'indentation à partir de la gauche

`\parshape` est utile lorsqu'on veut que le texte entoure une image. C'est une façon de déjouer le caractère rectangulaire du programme `TEX`. Par exemple, ce paragraphe a été écrit en utilisant `\parshape` avec les valeurs `8 0pt 8cm 0pt 6cm 0pt 4cm 0pt 2cm 0pt 4cm 0pt 6cm 0pt 8cm 0pt 12cm`

