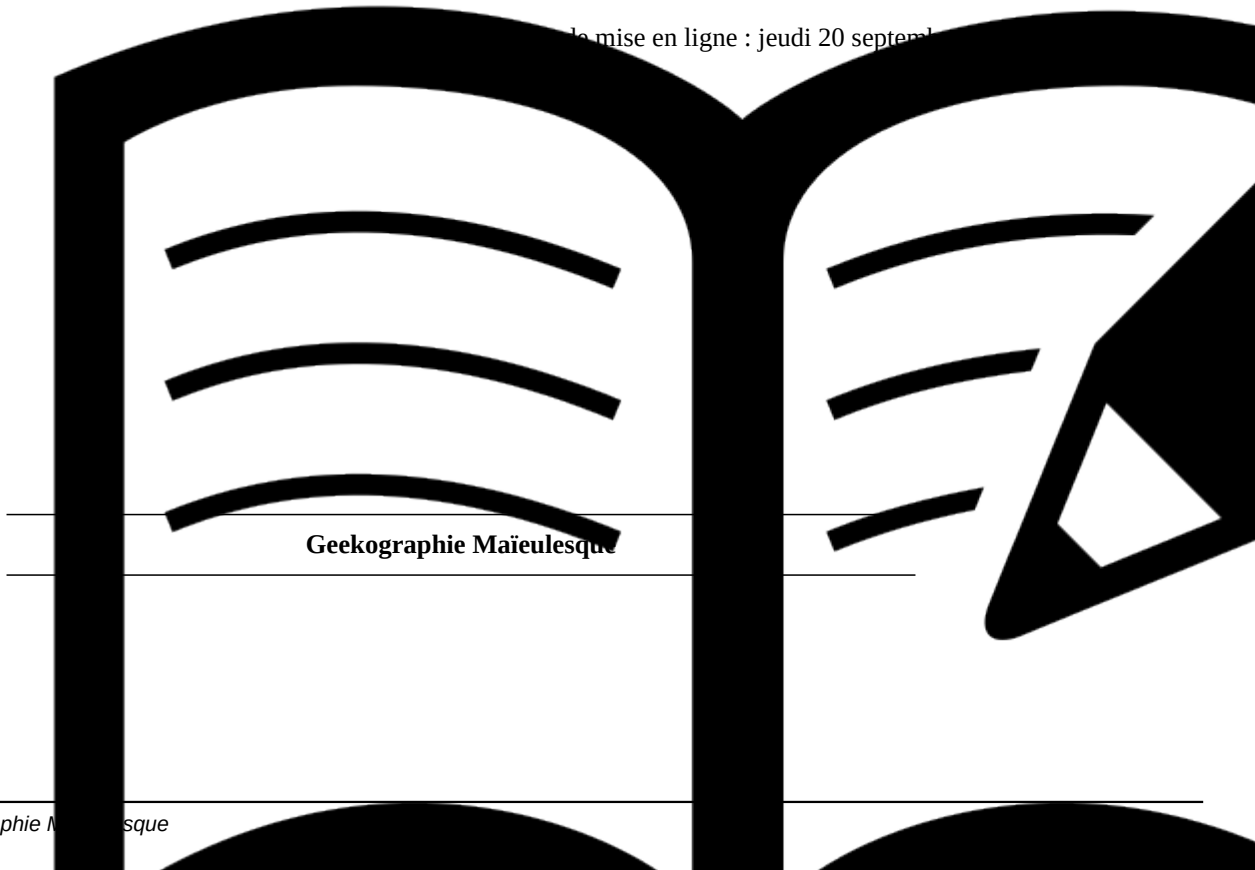
Extrait du Geekographie Maïeulesque

http://geekographie.maieul.net/Formaliser-les-variantes

Formaliser les variantes textuelles avec (E)ledmac

- LaTeX
- (e)led(mac/par)
-

Date de mise en ligne : jeudi 20 septembre



Geekographie Maïeulesque

Copyright © Geekographie Maïeulesque

Eledmac [1] permet d'associer des commentaires à certains mots d'un texte. Il peut donc servir pour établir des éditions critiques.

Néanmoins, il ne propose pas de formalisation de l'encodage des variantes. Chacun est donc libre d'utiliser son propre encodage.

Cet article décrit une manière de formaliser ces encodages en tirant tous le parti des commandes LaTeX, et notamment du package *etoolbox* [2].

Le présent article est désormais inutile, si ce n'est pour comprendre comment produire de telles commandes, puisque j'ai publié un package en reprenant le principe.

Notre exemple

Soit cinq manuscrits d'un texte : P, A, B, C, D. P est le manuscrit qui sert de référence. Les variantes de A, B, C, D sont indiquées dans l'apparat.

Soit le pseudo-mot « Lorem ». Ce mot subit les variantes suivantes :

- manuscrits A et B : mot omis.
- manuscrit C : mot remplacé par « loram ».
- manuscrit D : mot remplacé par « lorim ».

Approche non formalisée

Si on ne formalise notre apparat, on pourrait mettre :

```
\edtext{lorem}{\Afootnote{AB \emph{omit} ; C loram ; D lorim}}
```

Qui produit dans l'apparat :

> lorem] AB *omit* ; C loram ; D lorim

Les limites de cette approche sont évidentes :

- Si on change de présentation, on doit changer toutes nos notes.
- Impossible de produire des statistiques.
- Difficulté à exporter vers d'autres formats que LaTeX.

Copyright © Geekographie Maïeulesque

- Impossible de vérifier si on ne s'est pas trompé dans l'indication du manuscrit.

Proposition de formalisation

Pour obtenir le même résultat, nous proposons la formalisation suivante :

```
\var{lorem}{A,B}{
{{C}{loram}},
{{D}{lorim}}
}
```

`\var` est une commande personnalisée, qui prend les arguments suivants :

1. Texte principal
2. Liste des manuscrits où le mot est omis.
3. Liste des variantes, séparées par des virgule. Chaque variante est indiquée sous la forme `{manuscrit}{variante}`.

Mise en oeuvre : liste des commandess

Nous allons déclarer les commandes suivantes :

- `\var` (3 arguments), commande principal, qui appellera :
  - `\del` (1 argument), chargée d'indiquer les omissions [3]
  - `\variantes` (1 argument) chargé d'indiquer les variantes. Cette commande appelera elle même une commande `\variante` chargé de formaté une variante particulière (2 argument : le manuscrit et la variante).

Ce qui donne le schéma suivant :
<a href="IMG/png/formalisation.png" title='PNG - 10 ko' type="image/png">

Schéma des commandes d'indication de variantes

Mise en oeuvre, étape 1 : la commande `\del`

Commençons par déclarer notre commande principale, `\variante`.

```
\newcommand{\var}[3]{%
    \edtext{#1}{\Afootnote{\del{#2}}}
}
```

Pour le moment, nous ne nous occupons que d'indiquer les omissions. Nous passons le premier argument de `\var` comme premier argument de `\edtext`, et nous passons le second argument à la commande `\del`, à l'intérieur de l'argument dans `\Afootnote`.

La commande `\var` est définie ainsi :

```
\newcommand{\del}[1]{%
 \renewcommand{\do}[1]{##1}%
 \docsvlist{#1} \emph{omit}%
}
```

Analysons le code, en commençant par la ligne 3 :

`\docsvlist` est une commande d'etoolbox. Elle analyse une liste d'éléments séparés par des virgules [4]. Chaque élément de la liste est passé à une commande `\do`. Ici, nous analysons l'argument `#1`, qui contient la liste des manuscrits où la variante est omise.

Une fois cette boucle effectuée, nous indiquons que les manuscrits omettent la variante via `\emph{omit}`

La commande `\do` est définie en standard par etoolbox. Cependant nous allons ici la rédéfinir, mais uniquement à l'intérieur de `\del`. C'est à dire que la définition que nous en donnons ne sera pas valable si on l'appel en dehors de `\del`.

Ici, nous nous contentons d'afficher l'argument, dans le cas présent le manuscrits. Comme notre commande est définie à l'intérieure d'une autre commande, ses arguments sont indiqués par deux `#`, et non pas un seul.

Si nous utilisons la formulation précédente, nous obtenons alors :

> lorem] AB *omit*

Ce qui est bien. Néammoins supposons que nous ne souhaitons pas indiquer de manuscrits omettant le mot, mais seulement des manuscrits le changeant :

```
\var{ipsum}{}{{C}{ipsem}}
```

On obtient alors un inesthétique :

> ipsum] *omit*

Pour éviter ce problème, nous allons utiliser la commande `\ifstrempty{chaîne}{sioui}{sinon}`, qui appelle `sioui` si `chaîne` est vide, et `sinon` si la chaîne n'est pas vide.

Nous ferons l'appel dans la commande `\variante`, pour conditionnner l'appel à `\del` :

```
\newcommand{\var}[3]{%
   \edtext{#1}{\Afootnote{%
       \ifstrempty{#2}{}{\del{#2}}%
   }}
}
```

Mise en oeuvre, étape 2 : les commandes `\variante` et `\variantes`

La commande `\variantes` est la suivante :

```
\newcommand{\variantes}[1]{%
 \newif\iffirst%
 \firsttrue%
 \renewcommand{\do}[1]{\iffirst\firstfalse\else ; \fi\variante##1}%
 \docsvlist{#1}%
 }
```

Analysons :

- L. 2-3 : nous déclarons un test : `\iffirst`. Cela nous permettra d'afficher un point virgule avant le variante uniquement si nous ne somme pas à la première variante.
- L. 4 : à nous une commande `\do` :
  - `\iffirst\firstfalse\else ; \fi` : `\iffirst` teste si nous sommes à la première variante (*cf.* l. 2-3).
    - Si tel est le cas, nous appelons `\firstfalse`, qui permettra que `\iffirst` soit faux prochain passage.
    - Si tel n'est pas le cas (`\else`), on affiche un point virgule : `\else ;` .
    - `\fi` finit la structure de test commencé par `\iffirst`.

  - Nous appelons ensuite la commande `\variante`. Notez à nouveau le double `#`. Vous remarquerez que nous ne mettons pas de `{}` pour indiquer les arguments de la commande. En effet, si vous vous rappellez de la syntaxe utilisée, nos variantes sont justement indiqué sous la forme `{manuscrit}{texte}`.
    - L. 5 nous bouclons sur toute les variantes.

Passons à la commande `\variante`. Elle est déclaré ainsi :

```
 \newcommand{\variante}[2]{%
 #1 #2%
 }
```

Rien de bien méchant : on affiche simplement le manuscrit (`#1`, puis la variante (`#2`).

Il nous faut maintenant appeler `\variantes` à l'intérieur de `\var`.

```
 \newcommand{\var}[3]{%
    \edtext{#1}{\Afootnote{%
        \ifstrempty{#2}{}{\del{#2}\ifstrempty{#3}{}{ ; }}%
        \variantes{#3}%
    }}
}
```

Le seul élément notable est l. 3 : après avoir affiché les omissions, nous affichons un point virgule, si jamais nous avons des variantes (`\ifstrempty{#3}{}{ ; }`.

Nous obtenons alors le résultat désiré :

lorem] AB *omit* ; C loram ; D lorim

Et pour un peu plus de contrôle : une liste des manuscrits

Suppons que par inadvertance, nous ayons indiqué un manuscrit `E`, qui n'existe pas :

```
\var{dolor}{E}{}
```

Il serait bon que LaTex nous indique notre erreur.

Pour ce faire nous allons utiliser :

- le mécanisme d'affichage d'erreur dans les logs, fournit par eledmac, à savoir la commande `\eledmac@warning`.
- le mécanisme d'etoolbox de déclaration de liste.

Comme la commande `\eledmac@warning` contient un arobase, il nous faudra déclarer nos commande entre `\makeatletter` et `\makeatother`.

Commençons par créer une commande : `\manuscrits`. Cette commande contiendra une liste de manuscrits, qu'etoolbox pourra consulter pour :

- vérifier si un manuscrit existe, avec `\ifinlist{\manuscrits}{sioui}{sinon}`.
- faire une boucle dessus pour afficher les manuscrits avec `\dolistloop{\manuscrits}`.

```
 \newcommand{\manuscrits}{}
\listadd{\manuscrits}{A}
\listadd{\manuscrits}{B}
\listadd{\manuscrits}{C}
\listadd{\manuscrits}{D}
```

L. 1 nous déclarons une liste vide, l. 2-5 nous la complétons.

Puis modifions nos commandes `\del` et `\variante`.

```
 \newcommand{\del}[1]{%
 \renewcommand{\do}[1]{%
  \ifinlist{##1}{\manuscrits}%
  {##1}%
  {\eledmac@warning{Man. ##1 inconnu, p.\the\page@num  ; l.\the\line@num}Man. ##1 inconnu}%
 }%
 \docsvlist{#1} \emph{omit}%
}
```

- L. 3 : Nous testons si le manuscrit se trouve dans la liste
- L 4 : Si tel est le cas, nous l'affichons.
- L 5 : sinon nous émettons un message d'erreur, précisant le numéro de page (`\the\page@num` et de ligne `\the\line@num` [5]). Nous affichons également dans le PDF l'erreur.

Pour `\variante`, le code est similaire, mais nous indiquons également la variante proposée :

```
\newcommand{\variante}[2]{%
\ifinlist{#1}{\manuscrits}%
 {#1 #2}%
 {\eledmac@warning{Man. #1 inconnu, p.\the\page@num  ; l.\the\line@num ; #2}Man. #1 #2}%
}
```

Post-scriptum :

En guise d'exercice, par difficulté croissante :

1. *Remplacer le séparateur entre les variantes par une double barre verticale.*
2. *Créer une variante de* `\var` qui permette de déclarer une même variante pour plusieurs manuscrits : `\var{amet}{}{{A,B}{amet}}`
3. Créer une commande `\ledmanuscrits` qui reçoit comme argument une liste de manuscrits séparés par des virgules et les ajoute à `\manuscrits`.
4. Améliorer le code pour imposer que l'ordre des manuscrits soit celui indiqué dans `\manuscrits` (je n'ai pour le moment aucune idée de l'algorithmique derrière).

---

[1] Où ledmac, son précédesseur

[2] Ce package est chargé par défaut avec eledmac, mais non pas avec ledmac.

[3] Il n'est pas possible de déclarer `\omit` : une telle command existe déjà dans LaTeX.

[4] `csv` = « comma separated values »

[5] `\page@num` et `\line@num` sont des compteurs internes à eledmac. Néanmoins, ce sont des compteurs TeX et non pas LaTeX : d'où la syntaxe `\the\xxx` pour les afficher.

Copyright © Geekographie Maïeulesque