

Documentation française de l'extension `yagusylo`*

Le T_EXnicien de surface
le.texnicien.de.surface@wanadoo.fr

2009/02/26

Résumé

Cette extension permet d'obtenir un symbole sans avoir à charger l'extension qui le fournit habituellement. Cela permet d'éviter des conflits de noms.

On peut la considérer comme une version étendue de `pifont` en `technicolor`.

Abstract

The documentation of `yagusylo` is available in English with the name `yagusylo-en` and `yagusylo-en.pdf` should be available with this package.

Table des matières

1	Introduction	4
2	Noms spéciaux et conventions générales	4
I	Utilisation	6
3	Macros pour glyphe unique	6
3.1	Des clés et de leur réglage	6
3.2	Les macros elles-mêmes	6
3.3	Pour les initiés	7
4	Remplissage et ligne	8
4.1	Remplissage	8
4.1.1	Les clés	8
4.1.2	Macros de remplissage	8
4.1.3	Dans les coulisses	8
4.2	Tracer une ligne	9
4.2.1	Clés spéciales pour les lignes	9
4.2.2	Macros pour tracer une ligne	9
5	Listes	10
5.1	L'environnement <code>yagitemize</code>	10
5.1.1	L'environnement <code>yagitemize</code>	10
5.1.2	Et comment le régler	10
5.2	L'environnement <code>yagitemize*</code>	10
5.2.1	L'environnement <code>yagitemize*</code>	10
5.2.2	Et comment le régler	10

*Ce document correspond au fichier `yagusylo v1`, du 2009/02/26, édition du cinquantenaire.

6	Énumération	11
6.1	Clés spéciales pour <code>yagenumerate</code>	11
6.2	Les motifs et leur création	11
6.3	L’environnement <code>yagenumerate</code>	12
6.4	Définir l’environnement	12
7	Mélanges	12
7.1	Fichier de configuration	12
7.2	Les couleurs et comment s’en débarrasser	12
II	Résumé de l’utilisation	13
8	Les clés de l’extension	13
9	Commandes et environnements	13
9.1	Macros pour un glyphe	14
9.2	Remplissage et ligne	14
9.3	Itemize et <code>enumerate</code> à la <code>yagusylo</code>	15
III	Exemples	17
10	<code>\yagding</code>, <code>\defdingname</code> et <code>\yagding*</code>	17
11	<code>\yafgill</code> et <code>\yagfill*</code>	17
11.1	La clé « <code>leadtype</code> »	17
11.2	La clé <code>symplace</code>	18
11.2.1	Valeurs différentes de <code>a</code>	18
11.2.2	Valeur <code>a</code>	18
12	Environnements <code>yagitemize</code> et <code>yagitemize*</code>	19
12.1	Environnement <code>yagitemize*</code>	19
13	Fichier de configuration	20
IV	Galerie	21
14	De l’extension <code>pifont</code>	21
14.1	Les symboles de la <code>symfam pifont</code>	21
15	De l’extension <code>ifsym</code>	22
15.1	Les symboles de la <code>symfam ifsym</code>	22
15.2	Les symboles de la <code>symfam ifsymgeo</code>	23
15.3	Les symboles de la <code>symfam ifsymgeonarrow</code>	24
15.4	Les symboles de la <code>symfam ifsymgeowide</code>	25
15.5	Les symboles de la <code>symfam ifsymweather</code>	26
15.6	Les symboles de la <code>symfam ifsymclock</code>	27
16	De l’extension <code>marvosym</code>	28
16.1	Les symboles de la <code>symfam marvosym</code>	28
17	De l’extension <code>fourier</code>	29
17.1	Les symboles de la <code>symfam fourier</code>	29
18	De l’extension <code>wasysym</code>	30
18.1	Les symboles de la <code>symfam wasysym</code>	30

19 De l'extension <code>bbding</code>	31
19.1 Les symboles de la symfam <code>bbding</code>	31
20 De l'extension <code>dingbat</code>	32
20.1 Les symboles de la symfam <code>dingbat</code>	32
20.2 Symboles larges de <code>dingbat</code>	33
20.3 Les symboles de la symfam <code>ark</code>	34

1 Introduction

J’ai commencé à écrire cette extension lorsque, voulant placer le symbole ☒ devant mon numéro de téléphone dans un document utilisant la classe `lettre`, je me suis rendu compte que `marvosym` et `lettre.cls` définissaient tous deux une macro `\fax`. Du fait de ce conflit de noms, je fus très déçu.

De fait, très souvent, nous n’utilisons que quelques symboles parmi tous ceux fournis par une extension comme `marvosym`. Aussi un grand nombre de macros sont définies et chargées pour pas grand’ chose. `yagusylo` limite le nombre de macros mais ne peut empêcher \TeX de charger toutes les fontes nécessaires, ce qui est plutôt bien.

En fait, après réflexion, je n’ai certainement pas choisi la route la plus courte pour me sortir du bourbier où je m’étais enfoncé en me décidant à écrire cette extension. Cependant, comme elle était écrite, je n’aurais pas apprécié de la laisser prendre la poussière dans un coin reculé de mon disque dur et je l’ai chargé sur mon site comme une sorte de version pré-CTAN.

Puis, je me reposai quelques temps avant de charger la version d’alors — après quelques toilettes — sur le CTAN et je me retrouvai à relire un passage du *LaTeX Companion 2nd edition* et tombai sur le passage concernant l’extension `pifont`. Et je fus attéré car il me semblait que mon extension n’était plus qualifiée pour une rapide CTANification.

Je me décidai alors à ajouter quelques nouvelles capacités à `yagusylo` et tout d’abord, quelques équivalents aux environnements fournis par `pifont`. Aujourd’hui, après quelques travaux et un bon peu d’aide gracieusement fournie par les habitués de `fr.comp.text.tex`, à savoir Jean-Côme CHARPENTIER, Arnaud SCHMITTBUHL et Manuel PÉGOURIÉ-GONNARD, je peux sans trop rougir déposer la vraie première version de mon petit travail.

Avant de passer aux explications quant à son utilisation, je finirai par un petit mot au sujet de son nom. `yagusylo` est l’acronyme de *Yet another grand unified symbols loader* — Encore un grandiose chargeur unifié de symboles. On y trouvera certainement quelque ironie puisque je ne pense pas qu’il y ait beaucoup d’extension nommée *unified symbols loader* — chargeur unifié de symboles. Je laisse au lecteur le soin de décider si l’adjectif « grandiose » est vraiment approprié ; -)

2 Noms spéciaux et conventions générales

Désormais, une « famille de symboles » est un ensemble de glyphes qui est, dans les termes de NFSS, défini par un codage U, une famille et peut-être une série et une forme. `yagusylo` fournit des noms pour ces familles de symboles que l’on peut lire dans la table 1, page 13.

Il s’agit souvent du nom de l’extension qui fournit les symboles, p. ex. *fourier* ou *marvosym* mais, pour quelques extensions, il y a plus d’une famille : avec `dingbat` on a *dingbat* et *ark*.

Ainsi lorsque l’on demande la famille de symboles `marvosym`, cela revient à quelque chose comme `\fontencoding{U}\fontfamily{mvs}\fontseries{m}\fontshape{n}\selectfont` plus un peu de code pour ajouter de la couleur, si l’option `color` est choisie, et le fait que tout est fait dans un groupe pour limiter l’effet du changement de fonte.

Par pure fainéantise, j’utiliserai « symfam » comme abréviation de « famille de symboles ». Je ferai parfois référence à un symbole sous le nom de *ding*.

Cette extension utilise `xkeyval` pour manipuler ses options. Aussi une option est-elle en fait une clé et sa valeur. Il y a des options globales que l’on peut fixer dans le préambule dans l’argument optionnel de `\usepackage` p. ex.

```
\usepackage[onerror=nice, info=mute]{yagusylo}
```

qui fixe deux options globales, à savoir `onerror` et `mute`. Les clés des options globales sont rendues invalides à la fin du préambule aussi ne peut-on pas changer ces options dans le cours du document.

Les autres clés sont encore actives au début du document et sont utilisées intensivement pour modifier le comportement des macros de `yagusylo`. J’y ferai référence comme « options locales ».

Elles sont locales, premièrement, en ce que, en interne, je n'utilise pas `\gsetkeys` pour les fixer et, deuxièmement, en ce qu'elles « respectent les limites de groupes ». Aussi, si on règle quelque option locale dans un environnement, le réglage est confiné à cet environnement et l'option reprendra sa valeur précédente à la fin de l'environnement courant.

Toutes les clés locales affectent le comportement de presque toutes les commandes mais *pas* celui de l'environnement `yagenerate` qui dispose de ses propres clés.

Quelques macros ont une version étoilée et quelques unes même une version plussée. On trouvera donc à côté de `\unemacro`, `\unemacro*` et peut-être aussi `\unemacro+`. Là où la version nue `\unemacro` attend un $\langle num-car \rangle$ — un entier compris entre 0 et 255 — la version étoilée `\unemacro*` attend un $\langle nom-ding \rangle$ défini par les macros `\defdingname` or `\defdingname+`, cf. page 6.

Les versions plussées ne sont pas pour les cœurs de lièvres, ugh ! Elles attendent beaucoup d'arguments. Soyez-en conscient et évitez-les comme il se doit ; -)

J'écrirai `\unemacro(+)` pour faire référence aux deux macros `\unemacro` et `\unemacro+`, `\unemacro(*/+)` pour parler des trois commandes `\unemacro`, `\unemacro*` et `\unemacro+` ensemble.

Certaines macros acceptent un argument qui peut être une valeur « normale » ou la valeur *spéciale* `*`. J'écrirai cet argument sous la forme $\langle normal/* \rangle$.

Première partie

Utilisation

Remarque Dans ce document, `yagusylo` est chargé avec `\usepackage[color=true, onerror=nice]{yagusylo}`.

3 Macros pour glyphe unique

Les trois premières macros fournissent le moyen d'obtenir *un* glyphe. Leur comportement dépend, d'une certaine manière, de deux clés `symfam` et `symcolor` et c'est pourquoi je vais commencer par expliquer comment régler les clés dans `yagusylo`.

3.1 Des clés et de leur réglage

`\setyagusylokeys` On peut se servir de `\setyagusylokeys` avec pour seul argument obligatoire une liste de paires de la forme `clé=valeur` :

```
\setyagusylokeys{(liste de paires clé-valeur/*)}
```

Pour régler, p. ex., la clé `symfam` sur la valeur `marvosym` et la clé `symcolor` sur `gray` on tapera

```
\setyagusylokeys{symfam=marvosym, symcolor=gray}
```

et les valeurs seront fixées jusqu'à la fin du groupe dans lequel on a écrit les commandes ou jusqu'à l'utilisation suivante de `\setyagusylokeys`.

On peut utiliser `\setyagusylokeys` avec l'argument spécial `*` pour revenir aux valeurs par défaut des clés locales de l'extension.

Après `\setyagusylokeys{*}`, `symfam` a la valeur `pifont` et `symcolor` la valeur `red`, voir la table 2, page 13, pour une liste complète des clés locales et de leurs valeurs par défaut.

La macro `\setyagusylokeys` n'affecte pas le comportement de l'environnement `yagenu` `merate`.

3.2 Les macros elles-mêmes

`\yagding` `yagusylo` fournit la macro `\yagding` dont la syntaxe est :

```
\yagding[<famille>]{<num-car>}[<couleur>]
```

où `<famille>` est l'une des `symfams`. Par défaut, `<famille>` vaut `pifont` à moins que l'on ait donné une autre valeur à la clé `symfam`, dans le préambule ou dans le corps du document, avant d'utiliser `\yagding`.

Le `<num-car>` est le numéro du symbole dans le fichier de fonte qui le « décrit ». On trouvera en section IV, page 21, la liste de tous les symboles disponibles avec leurs famille et numéro. Dans tous les cas, `<num-car>` est un entier compris entre 0 et 255 inclusivement.

La `<couleur>` est le nom d'une couleur connue par `xcolor` qui se charge de tous les détails sordides de l'affaire. Par défaut, la couleur est la valeur de `symcolor` qui est elle-même `red` — rouge — par défaut.

Par exemple, j'obtiens « ❄ » avec `\yagding[fourier]{88}[blue]`. Avec `\yagding{88}`, j'obtiens « ❄ », symbole défini dans l'extension `pifont` avec la couleur rouge par défaut. Avec `\yagding{88}[green]`, j'obtiens « ❄ ».

Grâce à `xargs`, `yagusylo` fournit des macros qui acceptent plus d'un argument optionnel.

`\defdingname` Avec `\defdingname`, on peut donner un nom *local* ou *global* à un symbole :

```
\defdingname[<famille/*>][<defext>]{<num-car>}{<nom-ding>}[<couleur/*>]
```

où `<famille>`, `<num-car>` et `<couleur>` ont les mêmes significations que ci-dessus. De plus, `<famille>` et `<couleur>` ont aussi les mêmes valeurs par défaut que ci-dessus.

Si `<defext>` a la valeur `local` — qui est la valeur par défaut — le nom est *local* dans le sens que son existence est limitée au groupe courant. Pour obtenir une définition *globale*, on donnera la valeur `global` à `<defext>`. Toute autre valeur provoquera une erreur, si l'option `onerror` a la valeur `tough`, ou un avertissement et la définition sera *locale*.

Pour pouvoir utiliser le 2^e argument optionnel `<defext>`, on doit fournir le premier. On peut donc définir un nom comme suit :

`\defdingname[fourier][global]{116}{rhand}[red]`

et alors, même si la définition est faite dans un groupe, partout dans la suite du document on obtiendra «  » avec `\yagding*{rhand}`. Pour permettre à l'utilisateur d'obtenir le comportement habituel de `\defdingname` même avec un 2^e argument `global`, `yagusylo` autorise l'utilisation de `*` comme valeur du premier argument (optionnel). Aussi `\defdingname[*][global]{75}` créera, globalement, la macro que `\defdingname{75}` crée en local.

Le dernier argument (optionnel) *(couleur/*)* entraîne un comportement particulier quand sa valeur est `*`. Dans ce cas, la couleur du ding sera la couleur qui est la couleur courante au moment de l'utilisation de `\yagding*` et non pas, comme c'est le cas quand *(couleur/*)* n'est pas donné explicitement, celle courante au moment de la définition.

Pour être plus clair, supposons qu'à un certain point l'option `symcolor` ait pour valeur courante `red` et que l'on écrive

`\defdingname[fourier]{116}{hand}\defdingname[fourier]{116}{handvar}[*]`

alors tant que l'on ne change pas l'option `symcolor`, `\yagding*{hand}` et `\yagding*{handvar}` donneront le même glyphe  *mais* après

`\setyagusylokeys{symcolor=blue}`

si `\yagding*{hand}` donne toujours , `\yagding*{handvar}` donne .

Remarque Les macros de `yagusylo` dont le nom commence par `def`, comme `\defdingname`, ne vérifient pas l'existence préalable et permettent la redéfinition.

`\yagding*` On pourra alors utiliser `\yagding*` pour obtenir le symbole nommé avec `\yagding*{<nom-ding>}`

et le glyphe obtenu n'est pas affecté par les réglages des clés locales sauf `symcolor` dans le cas spécial d'une définition utilisant `*` pour 4^e argument comme expliqué ci-dessus, cf. page 7.

Par exemple, avec `\defdingname[fourier]{116}{doigt}[gray]` on définit le nom du symbole «  » et on peut l'obtenir ensuite avec `\yagding*{doigt}`.

En fait, le **vrai** nom de la macro utilisée en interne par `yagusylo` est `\Y@G@_doigt`. Si, avec un tel nom, il y avait encore un conflit de nom, c'est que quelqu'un l'aura fait exprès.

3.3 Pour les initiés

`\yagding+` La syntaxe de `\yagding+` est :

`\yagding+{<codage>}{<famille>}{<série>}{<forme>}{<num-car>}{<couleur>}`

où *<num-car>* et *<couleur>* ont les significations définies ci-dessus. La valeur par défaut de l'argument optionnel *<couleur>* est la couleur courante, comme ci-dessus encore.

Tous les autres arguments font référence aux spécification de la NFSS : *<codage>* est le codage de la fonte, dont la valeur par défaut est `U`, *<famille>* est la famille de la fonte — *family* —, *<série>* est la série de la fonte — *series* — et *<forme>* est sa forme — *shape*. Si l'on veut utiliser un glyphe dont la série ou la forme sont indéfinies, on donnera la valeur `*` à l'argument.

Ainsi `\yagding+{futs}{*}{*}{84}[blue]` donne «  », `\yagding+{futs}{*}{*}{85}` donne «  ».

`\defdingname+` La syntaxe de `\defdingname+` est :

`\defdingname+{<codage>}{<defext>}{<famille>}{<série>}{<forme>}{<num-car>}{<nom-ding>}{<couleur/*>}`

où les arguments *<nom-ding>*, *<defext>* et *<couleur/*>* ont le même rôle que dans `\defdingname`.

Par défaut *<codage>*, qui attend un codage de fonte, a pour valeur `U`; *<série>* et *<forme>* ont des valeurs qui les font ignorer. *<famille>* apprécierait de recevoir le nom légal d'une famille de fonte.

Une fois défini le nom du symbole, on peut l'utiliser avec le `\yagding*` habituel.

Attention! Pour utiliser un codage de fonte autre que `U`, on doit le déclarer dans le préambule comme argument de `fontenc`. Afin de pouvoir montrer ce qui suit, j'ai ceci `\usepackage[T2C,T1]{fontenc}` en préambule de ce document, ce qui fait que `\yagding+[T2C]{cmr}{m}{n}{128}[blue]` donne .

4 Remplissage et ligne

Les macros `\yagline(*/+)` utilisent `\yagfill(*/+)` pour placer les dings aussi je commence par le remplissage — *filling*.

4.1 Remplissage

Le mécanisme de remplissage est basé sur les commandes T_EX `\leaders`, `\xleaders` et `\cleaders`. Quelques clés sont réservées aux réglages du comportement de `\yagfill(*/+)` et gouvernent aussi celui de `\yagline(*/+)`.

4.1.1 Les clés

Six clés d'option gouvernent le comportement des macros `\yagfill(*/+)` : `leadtype`, `symplace`, `sympos`, `boxwidth`, `before` et `after`.

`leadtype` La clé `leadtype` a la valeur `l` par défaut et peut prendre les valeurs `x` or `c`. Avec `l` la macro `\leaders` de T_EX est utilisée, avec `c` c'est `\cleaders` et avec `x` c'est `\xleaders`. Quelques exemples montrent les différences d'aspect, cf. page 17.

`symplace` La clé `symplace` prend une valeur parmi `c` — valeur par défaut —, `r`, `l`, `a` et `n`.

`sympos` Si on choisit `n` alors on doit régler la clé `sympos`. Elle demande un entier compris entre 0 et 1 000, bornes comprises.

`before` Si on choisit `a`, les clés `before` et `after` doivent être définies. Ces deux options attendent une longueur L^AT_EXienne positive ou nulle. Si l'on ne fixe pas explicitement la valeur de `after`, elle prendra celle de `before`.

`boxwidth` Sauf quand on choisit `a`, `boxwidth` doit prendre une longueur L^AT_EXienne positive ou nulle. Si `boxwidth` a pour valeur `0 pt` — ou toute autre longueur nulle — alors la largeur effective de la boîte sera la largeur naturelle du symbole¹ utilisé par la macro `\yagfill(*)`. En fait, cela sera ainsi chaque fois que la valeur donnée à `boxwidth` sera inférieure à la largeur naturelle du symbole.

4.1.2 Macros de remplissage

`\yagfill` La syntaxe de `\yagfill` est la suivante :

`\yagfill`[(*liste de paires clé-valeur*)]{(*num-car*)}

où la (*liste de paires clé-valeur*), si elle est fournie, règle les valeurs des clés de la liste. Au cas où la valeur d'une clé n'est pas donnée explicitement, elle conserve sa valeur courante. Ainsi, si on ne donne pas cet argument optionnel, les clés suivantes ont leur valeur courante : `symfam`, `symcolor`, `leadtype`, `symplace`, `sympos`, `boxwidth`, `before` et `after`.

(*num-car*) a la même signification que pour `\yagding` ci-dessus.

`\yagfill*` La syntaxe de `\yagfill*` est la suivante :

`\yagfill*`[(*liste de paires clé-valeur*)]{(*nom-ding*)}

où la (*liste de paires clé-valeur*) a la même utilisation que dans `\yagfill` mais où (*nom-ding*) doit être le nom d'un symbole défini préalablement à l'aide de `\defdingname(+)`.

`\yagfill+` La syntaxe de `\yagfill+` est la suivante :

`\yagfill+`[(*liste de paires clé-valeur*)]{(*matériel*)}

où (*matériel*) est quelque chose qui peut être composé et a une largeur positive.

On trouvera quelques exemples en section 11, page 17.

4.1.3 Dans les coulisses

Les trois macros utilisent le même code interne pour fabriquer les boîtes utilisées par la macro T_EXienne `\leaders`, `\cleaders` ou `\xleaders`. La macro généraliste est évidemment la version plussée.

D'abord, lorsque `symplace` ne vaut pas `a`, la boîte a pour largeur `boxwidth` a moins que cette dernière valeur soit inférieure à la largeur naturelle de la boîte contenant le matériel à composer, auquel cas on prend cette largeur naturelle.

1. Cette largeur naturelle est obtenue avec `\settowidth`.

5 Listes

yagusylo fournit deux environnements `yagitemize` et `yagitemize*` et deux macros `\setyagitemize(*)` pour en fixer le comportement par défaut.

5.1 L'environnement `yagitemize`

5.1.1 L'environnement `yagitemize...`

Voici le 1^{er} environnement de yagusylo. Comme son nom peut l'indiquer, c'est une sorte d'environnement `itemize`. Sa syntaxe est la suivante

```
\begin{yagitemize}[\langle symfam \rangle]{\langle num-car/* \rangle}[\langle couleur \rangle]
puis viennent quelques \items et enfin
\end{yagitemize}
```

comme d'habitude.

L'argument obligatoire $\langle num-car \rangle$ doit être un numéro de caractère comme défini précédemment ou une étoile `*` auquel cas le comportement de l'environnement change quelque peu : `yagitemize` compte sur des valeurs par défaut que l'on devra avoir fournies préalablement à l'aide de `\setyagitemize` que je présenterai plus loin.

On peut enboîter autant de `yagitemize` que l'on veut mais je décline toute responsabilité quant à l'esthétique. De plus, comme `yagitemize` repose sur l'environnement `list` bien connu et présent partout, L^AT_EX pourrait se plaindre d'un trop grand nombre de listes emboîtées. N'oublions pas que, p. ex., un environnement `quote` est aussi une `list`.

5.1.2 Et comment le régler

La macro `\setyagitemize` permet de définir différents symboles pour différents niveaux d'emboîtement de l'environnement `yagitemize`. Il ne prend qu'un argument obligatoire qui doit avoir la forme suivante

```
symfam1, nombre1, couleur1. symfam2, nombre2, couleur2. ... symfamn, nombren,
couleurn
```

On change de niveau avec un point « . ». Pour chaque niveau, on doit fournir trois valeurs séparées par des virgules. La première est une *symfam* comme définie ci-avant, la deuxième est le numéro du symbole demandé, la troisième est la couleur de ce symbole. Je n'ai pas fourni de mécanismes utilisant des valeurs par défaut : chacun de ces trois arguments doit être donnés explicitement.

Lorsque `yagitemize` atteint le niveau $n + 1$, où n est le numéro du dernier triplet fourni, il émet un avertissement ou une erreur, suivant la valeur de la clé globale `onerror`, et, si `onerror` n'a pas la valeur *tough*, il continue avec le réglage du n -ième niveau.

Je me suis arrangé pour que le premier `yagitemize` utilise le premier réglage donné dans `\setyagitemize`, — au prix de la définitions de quelques compteurs L^AT_EXiens — quel que soit le niveau d'emboîtement de listes auquel on se place.

5.2 L'environnement `yagitemize*`

5.2.1 L'environnement `yagitemize*...`

`yagitemize*` L'environnement `yagitemize*` a la syntaxe suivante :

```
\begin{yagitemize*}[\langle nom-ding \rangle]
```

son argument optionnel, s'il est fourni, doit être le nom d'un symbole, comme il est habituel pour une macro étoilée. Lorsque l'on ne fournit aucun argument $\langle nom-ding \rangle$, l'environnement utilise le réglage par défaut obtenu à l'aide de `\setyagitemize*`.

5.2.2 Et comment le régler

La macro `\setyagitemize*` permet de définir différents symboles pour les différents niveaux d'emboîtement des environnements `yagitemize*`. Elle ne prend qu'un seul argument obligatoire qui doit avoir la forme suivante.

```
dingname1. dingname2. ... . dingnamen
```

où chaque `dingname k` doit être un nom valide défini par `\defdingname(+)`, cf. page 19, exemple 8.

Le mécanisme en est analogue à celui de `\setyagitemize` et `\setyagitemize*` a sur l’environnement `yagitemize*` le même effet que la macro non-étoilée sur l’environnement non-étoilé.

Remarque Je ne fournis pas de version plussée de cet environnement car il est toujours possible de définir un nom de symbole avec `\defdingname+` et de l’utiliser dans `\setyagitemize*`.

6 Énumération

De même que `yagitemize` singe le `itemize` de L^AT_EX, `yagenumerate` copie `enumerate` mais avec un costume `yagusylo`-esque.

6.1 Clés spéciales pour `yagenumerate`

Le comportement de l’environnement `yagenumerate` est contrôlé par les clés suivantes : `symfam`, `symcolor`, `firstitemnum`, `enumlah` et `enumpattern`. Les deux clés `symfam` et `symcolor`, dans ce contexte, sont différentes des *clés locales ordinaires* `symfam` et `symcolor`. On pourrait dire qu’il y a deux trousseaux de clés, un pour `yagenumerate` — connu comme le *trousseau enum* — et l’autre — appelé *trousseau général* — pour tout le reste et que, même si elles se ressemblent, deux clés attachées à des trousseaux différents n’ouvrent pas les mêmes portes.

<code>symfam</code> [enum]	Toutefois, les clés <code>symfam</code> et <code>symcolor</code> du trousseau <code>enum</code> contrôlent effectivement la <code>symfam</code> et la couleur des dings utilisés dans l’énumération. Au début du document — après le <code>\begin{document}</code> — elles ont, respectivement, pour valeur <code>pifont</code> et <code>blue</code> .
<code>symcolor</code> [enum]	
<code>firstitemnum</code>	Dans un environnement <code>yagenumerate</code> , chaque <code>\item</code> incrémente un compteur qui pointe sur le glyphe utilisé pour cet article. Le numéro, comme d’habitude dans la <code>symfam</code> , du premier glyphe est fixé par <code>firstitemnum</code> qui vaut par défaut 172.
<code>enumlah</code>	La clé <code>enumlah</code> fixe le nombre d’articles pouvant apparaître au même niveau de <code>yagenumerate</code> . Sa valeur par défaut est 10. Au delà, on aura une erreur quelle que soit la valeur de la clé <code>onerror</code> .
<code>enumpattern</code>	La clé spéciale <code>enumpattern</code> est encore plus spéciale. J’en donne une explication détaillée dans la section suivante.

6.2 Les motifs et leur création

Un `pattern` — motif — pour `yagenumerate` est un moyen de conserver et appeler un ensemble complet de clés spéciales.

Il y a quatre motifs prédéfinis, à savoir `piwcr`, `piwcs`, `pibcr` et `pibcs`. Le motif par défaut est `piwcr`. Dans ces noms `pi` signifie `pifont` ; `c` est là pour `circle` ; `w` pour `white` et `b` pour `black` ce qui est peut-être maladroit comme on le verra bientôt ; `s` est là pour `sans-serif` et `r` pour `roman`. Ils limitent tous la longueur de l’énumération à 10.

Voici le premier nombre de chacun de ces motifs :

- ☞ `piwcr` : ①
- ☞ `piwcs` : ①
- ☞ `pibcr` : ①
- ☞ `pibcs` : ①

<code>\newenumpattern</code>	On peut définir son propre motif avec <code>\newenumpattern</code> dont la syntaxe est : <code>\newenumpattern{<patname>}{<liste de paires clé-valeur>}</code> où <code><patname></code> est le nom du motif et où <code><liste de paires clé-valeur></code> contient au moins <code>symfam</code> , <code>firstitemnum</code> et <code>enumlah</code> . Si l’on n’utilise pas <code>symcolor</code> , la couleur du motif sera la couleur courante au moment de la définition. Je n’ai pas prévu de mécanisme semblable à celui de <code>\defdingname</code> mais on peut me le demander si l’on n’en éprouve le besoin.
------------------------------	---

6.3 L'environnement `yagenerate`

`yagenerate`

L'environnement `yagenerate` commence avec

```
\begin{yagenerate}[\langle liste de paires clé-valeur/* \rangle]
```

et à l'intérieur on utilise `\item` comme il est habituel en \LaTeX .

S'il n'y a pas d'argument, c.-à-d. si l'on a saisi quelque chose comme

```
\begin{yagenerate}
```

```
\item ...
```

alors l'aspect de l'énumération est déterminée par les valeurs courantes de `symfam`, `symcolor`, `firstitemnum` et `enumlength`.

Si l'argument est `*`, l'aspect est déterminé par le motif par défaut courant.

Enfin on peut fixer l'aspect à l'aide d'une liste de paires clé-valeur. Les clés qui ne sont pas fournies ont alors leur valeur par défaut.

Comme `yagenerate` redéfinit `\item`, on ne peut utiliser un `enumerate` normal imbriqué dans un environnement `yagenerate` sans utiliser l'environnement `notyagenum` comme une sorte d'enveloppe du `enumeratede` \LaTeX , cf. page 20, exemple 9.

Les limites d'imbrication sont celles de \LaTeX .

6.4 Définir l'environnement

`\setyageneratekeys`

Pour régler les clés qui gouvernent l'aspect de `yagenerate`, on peut utiliser la macro `\setyageneratekeys` dont la syntaxe est analogue à celle de `\setyagusylokeys`, cf. page 6.

Avec `\setyageneratekeys{*}`, les clés `symfam`, `symcolor`, `firstitemnum`, `enumlength` et `enumpattern` retrouvent leurs valeurs par défaut.

7 Mélanges

Je place ici quelques sujets que je n'ai pas été capable d'introduire de manière pertinente jusque maintenant.

7.1 Fichier de configuration

`configfile`

On peut utiliser un fichier de configuration. `yagusylo` lira le fichier `yagusylo.cfg` si l'on a donné la valeur `true` à la clé `configfile`. Sa valeur par défaut est `false`. \TeX doit pouvoir trouver le fichier `yagusylo.cfg` sinon on aura une erreur.

7.2 Les couleurs et comment s'en débarrasser

`color`

Comme je l'écris plus haut, la gestion des couleurs est laissée à `xcolor` si la clé `color` est réglée, globalement, sur `true`. Cela fournit deux moyens de repasser en noir et blanc.

La première méthode consiste simplement à changer la valeur de `color` en `false`. Toutes les couleurs de `yagusylo` seront alors supprimées.

La seconde méthode est de passer l'option `monochrome` à `xcolor`. Pour ce faire, on chargera `yagusylo` comme suit :

```
\usepackage[color=true, XcolorOptions=monochrome]{yagusylo}
```

`XcolorOptions`

Je profite de ce que je mentionne `XcolorOptions` pour ajouter ceci : si l'on veut passer plus d'une option à `xcolor`, on doit en placer la liste entre accolades comme ceci :

```
\usepackage[color=true, XcolorOptions={monochrome, table}]{yagusylo}
```

Deuxième partie

Résumé de l'utilisation

8 Les clés de l'extension

La table 1 donne la liste de toutes les symfams connues à ce jour par yagusylo, ces symfams sont les valeurs possibles de la clé `symfam`.

extension	symfam	extension	symfam
<code>pifont</code>	<code>pifont</code>	<code>marvosym</code>	<code>marvosym</code>
<code>ifsym</code>	<code>ifsym</code>	<code>fourier</code>	<code>fourier</code>
	<code>ifsymgeo</code>	<code>wasysym</code>	<code>wasysym</code>
	<code>ifsymgeonarrow</code>	<code>bbding</code>	<code>bbding</code>
	<code>ifsymgeowide</code>	<code>dingbat</code>	<code>dingbat</code>
	<code>ifsymweather</code>		<code>ark</code>
	<code>ifsymclock</code>		

TABLE 1 – Les symfams

La table 2 montre toutes les clés d'option, leurs valeurs par défaut et toutes les autres valeurs possibles. Bien entendu, lorsque j'écris « n'importe quelle longueur », on doit comprendre que cette longueur doit avoir du sens dans le contexte de son utilisation.

clé	valeur par défaut	autres valeurs possibles
Clés d'options globales		
<code>info</code>	<code>normal</code>	<code>verbose</code> , <code>mute</code>
<code>onerror</code>	<code>tough</code>	<code>nice</code>
<code>color</code>	<code>false</code>	<code>true</code>
<code>XcolorOptions</code>		liste d'options connues par <code>xcolor</code>
<code>configfile</code>	<code>false</code>	<code>true</code>
Clés d'options locales, trousseau général		
<code>symfam</code>	<code>pifont</code>	voir la table 1
<code>symcolor</code>	<code>red</code>	toute couleur connue de <code>xcolor</code>
<code>leadtype</code>	<code>l</code>	<code>c</code> , <code>x</code>
<code>symplace</code>	<code>c</code>	<code>l</code> , <code>r</code> , <code>a</code> , <code>n</code>
<code>sympos</code>	<code>0</code>	entier entre 0 et 1 000 bornes comprises
<code>boxwidth</code>	<code>0,2 in</code>	n'importe quelle longueur positive
<code>before</code>	<code>0 pt</code>	n'importe quelle longueur positive
<code>after</code>	<code>0 pt</code>	n'importe quelle longueur positive
<code>head</code>	<code>36,135 pt</code>	n'importe quelle longueur
<code>tail</code>	<code>36,135 pt</code>	n'importe quelle longueur
Clés d'options locales, trousseau enum		
<code>firstitemnum</code>	<code>172</code>	entier entre 0 et 255 bornes comprises
<code>enumlength</code>	<code>10</code>	entier
<code>symcolor</code>	<code>blue</code>	toute couleur connue de <code>xcolor</code>
<code>symfam</code>	<code>pifont</code>	voir la table 1

TABLE 2 – Clés de yagusylo

9 Commandes et environnements

Je donne ici toutes les utilisations possibles des commandes et environnements de yagusylo.

J'utiliserai les *<denomination>*s suivantes pour faire référence à quelques objets bien définis :

- ♠ $\langle num-car \rangle$: un entier entre 0 et 255 bornes comprises ;
- ℏ $\langle nombre \rangle$: un entier pour lequel on peut fournir quelques propriétés supplémentaires ;
- ♠ $\langle symfam \rangle$: le nom symbolique de la symfam comme donné dans la table 1 ;
- ♠ $\langle couleur \rangle$: le nom symbolique d'une couleur connue de xcolor ;
- ⊞ $\langle nom-ding \rangle$: le nom d'un ding défini avec `\defdingname(+)` ;
- ♠ $\langle defext \rangle$: l'étendue de la définition, peut être `local` — valeur par défaut — ou `global` ;
- ♠ $\langle G-liste \rangle$: une liste composée d'un nombre quelconque de paires clé-valeur dans lesquelles les clés sont attachées au trousseau général ; cf. page 13,
- ♠ $\langle E-liste \rangle$: une liste composée d'un nombre quelconque de paires clé-valeur dans lesquelles les clés sont attachées au trousseau enum ; cf. page 13,
- ♠ $\langle longueur \rangle$: n'importe quelle longueur L^AT_EXienne ;
- ♠ $\langle longueur pos. \rangle$: n'importe quelle longueur L^AT_EXienne positive.

Au passage, l'énumération précédente, à la yagusylo, est obtenue avec

```
\begin{yagenumerate}[symfam=wasysym, firstitemnum=88, enumlength=14, sym
color=purple]
```

9.1 Macros pour un glyphe

```
\yagding{⟨num-car⟩}
\yagding{⟨num-car⟩}[⟨couleur⟩]
\yagding[⟨symfam⟩]{⟨num-car⟩}
\yagding[⟨symfam⟩]{⟨num-car⟩}[⟨couleur⟩]
\yagding{⟨nom-ding⟩}
```

```
\yagding+[⟨codage⟩]{⟨famille⟩}{⟨série/*⟩}{⟨forme/*⟩}{⟨num-car⟩}[⟨couleur⟩]
```

où $\langle codage \rangle$ est un codage de fonte (défaut U), $\langle famille \rangle$ une famille de fonte, $\langle série \rangle$ une série de fonte — utiliser * pour ne fournir aucune série —, $\langle forme \rangle$ une forme, *shape*, de fonte — utiliser * pour ne fournir aucune forme.

```
\defdingname{⟨num-car⟩}{⟨nom-ding⟩}
\defdingname[⟨symfam⟩]{⟨num-car⟩}{⟨nom-ding⟩}
\defdingname[*]{⟨num-car⟩}{⟨nom-ding⟩}
\defdingname[⟨symfam⟩][⟨defext⟩]{⟨num-car⟩}{⟨nom-ding⟩}
\defdingname[*][⟨defext⟩]{⟨num-car⟩}{⟨nom-ding⟩}
\defdingname{⟨num-car⟩}{⟨nom-ding⟩}[⟨couleur⟩]
\defdingname[⟨symfam⟩]{⟨num-car⟩}{⟨nom-ding⟩}[⟨couleur⟩]
\defdingname[*]{⟨num-car⟩}{⟨nom-ding⟩}[⟨couleur⟩]
\defdingname[⟨symfam⟩][⟨defext⟩]{⟨num-car⟩}{⟨nom-ding⟩}[⟨couleur⟩]
\defdingname[*][⟨defext⟩]{⟨num-car⟩}{⟨nom-ding⟩}[⟨couleur⟩]
\defdingname{⟨num-car⟩}{⟨nom-ding⟩}[*]
\defdingname[⟨symfam⟩]{⟨num-car⟩}{⟨nom-ding⟩}[*]
\defdingname[*]{⟨num-car⟩}{⟨nom-ding⟩}[*]
\defdingname[⟨symfam⟩][⟨defext⟩]{⟨num-car⟩}{⟨nom-ding⟩}[*]
\defdingname[*][⟨defext⟩]{⟨num-car⟩}{⟨nom-ding⟩}[*]
```

```
\defdingname+[⟨enc⟩][⟨defext⟩]{⟨famille⟩}{⟨série⟩}
{⟨forme⟩}{⟨num-car⟩}{⟨nom-ding⟩}[⟨couleur/*⟩]
```

```
\setyagusylokeys{⟨G-liste⟩}
\setyagusylokeys{*}
```

9.2 Remplissage et ligne

```
\yagfill{⟨num-car⟩}
\yagfill[⟨G-liste⟩]{⟨num-car⟩}
```

```
\yagfill*{\langle nom-ding \rangle}
\yagfill*[\langle G-liste \rangle]{\langle nom-ding \rangle}
```

```
\yagfill+{\langle matériel \rangle}
\yagfill+[\langle G-liste \rangle]{\langle matériel \rangle}
```

où $\langle matériel \rangle$ est quelque chose qui peut être composé et a une largeur positive. **Attention** : on ne s'attendra pas à ce que cette macro fonctionne avec n'importe quoi !

```
\setyagline{\langle longueur \rangle}
\setyagline[\langle longueur \rangle]{\langle longueur \rangle}
```

```
\yagline{\langle num-car \rangle}
\yagline[\langle G-liste \rangle]{\langle num-car \rangle}
\yagline*{\langle nom-ding \rangle}
\yagline*[\langle G-liste \rangle]{\langle nom-ding \rangle}
\yagline+{\langle matériel \rangle}
\yagline+[\langle G-liste \rangle]{\langle matériel \rangle}
```

9.3 Itemize et enumerate à la yagusylo

Tous les environnements sont basés sur l'environnement `list`. On utilisera `\item` à l'intérieur pour obtenir une présentation intéressante mais cela vous regarde. Je ne donne la syntaxe que pour le début de l'environnement car je pense que l'on sait comment les clore ; -)

```
\begin{yagitemize}{\langle num-car \rangle}
\begin{yagitemize}{*}
\begin{yagitemize}[\langle symfam \rangle]{\langle num-car \rangle}
\begin{yagitemize}[\langle symfam \rangle]{*}
\begin{yagitemize}{\langle num-car \rangle}[\langle couleur \rangle]
\begin{yagitemize}{*}[\langle couleur \rangle]
\begin{yagitemize}[\langle symfam \rangle]{\langle num-car \rangle}[\langle couleur \rangle]
\begin{yagitemize}[\langle symfam \rangle]{*}[\langle couleur \rangle]
```

```
\begin{yagitemize*}
\begin{yagitemize*}[\langle nom-ding \rangle]
```

```
\setyagitemize{\langle liste spéciale \rangle}
```

avec

```
\langle liste spéciale \rangle = \langle triplet \rangle . \dots \langle triplet \rangle . \langle triplet \rangle
```

où

```
\langle triplet \rangle = \langle symfam \rangle , \langle num-car \rangle , \langle couleur \rangle
```

avec le sens habituel de $\langle symfam \rangle$, $\langle num-car \rangle$ et $\langle couleur \rangle$.

```
\setyagitemize* {\langle liste de noms-dings \rangle}
```

avec

```
\langle liste de noms-dings \rangle = \langle nom-ding \rangle . \dots \langle nom-ding \rangle . \langle nom-ding \rangle
```

avec le sens habituel de $\langle nom-ding \rangle$.

```
\begin{yagenumerate}
\begin{yagenumerate}[\langle E-liste \rangle]
\begin{yagenumerate}{*}
```

```
\setyagenumeratekeys{\langle E-liste \rangle}
\setyagenumeratekeys{*}
```

`\newenumpattern{patname}{E-liste}`

où *patname* est un nom que l'on peut utiliser ensuite comme valeur de la clé `enumpattern`.

Troisième partie

Exemples

10 \yagding, \defdingname et \yagding*

```

24 \setyagusylokeys{symfam=fourier}
25 \begin{quote}
26 dans un environnement \texttt{quote}\par
27 \defdingname{116}{lHand}\yagding*{lHand}
28 \quad
29 \defdingname{116}{lHandStar}[*]\yagding*{lHandStar}
30 \quad
31 \defdingname[*][global]{116}{gHandRed}[red]\yagding*{gHandRed}
32 \quad
33 \defdingname[*][global]{116}{gHandStar}[*]\yagding*{gHandStar}
34 \quad \yagding{117}
35
36 \setyagusylokeys{symcolor=blue}
37 \yagding*{lHand}\quad\yagding*{lHandStar}\quad
38 \yagding*{gHandRed}\quad\yagding*{gHandStar}\quad \yagding{117}
39 \end{quote}
40 hors environnement \texttt{quote}\par
41 \yagding*{lHand}\quad\yagding*{lHandStar}\quad
42 \yagding*{gHandRed}\quad\yagding*{gHandStar}\quad \yagding{117}
43
44 \setyagusylokeys{symcolor=green, symfam=pifont}
45 \yagding*{lHand}\quad\yagding*{lHandStar}\quad
46 \yagding*{gHandRed}\quad\yagding*{gHandStar}\quad \yagding{117}

```

dans un environnement quote





hors environnement quote

[? lHand ?] [? lHandStar ?]   

 [? lHand ?] [? lHandStar ?]   

```

28 \yagding+{logo}{m}{n}{77}[blue]\yagding+{logo}{m}{n}{69}[red]%
29 \yagding+{logo}{m}{n}{84}[gray]\yagding+{logo}{m}{n}{65}[black]%
30 \yagding+{logo}{m}{n}{80}[orange]\yagding+{logo}{m}{n}{79}[purple]%
31 \yagding+{logo}{m}{n}{83}[brown]\yagding+{logo}{m}{n}{84}[green]

```

METAPOST

11 \yafgill et \yagfill*

11.1 La clé « leadtype »

```

13 thingummy\yagfill{84}kinda big%
14 \setyagusylokeys{boxwidth=2cm,symcolor=gray}\par
15 thingummy\yagfill{84}kinda big\par
16 thing\yagfill{84}kinda very very big\par

```


The Squire of High Potternews	*	*	*	*	*	*	Jurisdiction
The Squire of High Potternews	*		*		*		Jurisdiction
The Squire of High Potternews	*	*		*		*	Jurisdiction
The Squire of High Potternews	*		*		*		Jurisdiction

```

_____ 7 — \yagfill+ _____
15 \defdingname[fourier]{116}{mainv}[green]
16 \defdingname[fourier]{116}{mainb}[blue]
17 \defdingname[fourier]{116}{mainr}[red]
18 \yagfill+[boxwidth=1.25cm]{\yagding*{mainv}%
19 \yagding*{mainb}\yagding*{mainr}}

```



12 Environnements yagitemize et yagitemize*

12.1 Environnement yagitemize*

```

_____ 8 — \yagitemize* et \setyagitemize* _____
18 \defdingname[fourier]{116}{mainv}[green]
19 \defdingname[fourier]{116}{mainb}[blue]
20 \defdingname[fourier]{116}{mainr}[red]
21 \defdingname[fourier]{116}{maing}[gray]
22 \setyagitemize*{mainv. mainb. mainr. maing}
23 \begin{yagitemize*}\item A\begin{yagitemize*}\item B
24 \begin{yagitemize*}\item C\begin{yagitemize*}\item D
25 \begin{yagitemize*}\item E
26 \begin{yagitemize*}\item F \item G \end{yagitemize*}
27 \item H\end{yagitemize*} \item I
28 \end{yagitemize*} \item J \end{yagitemize*} \item K
29 \end{yagitemize*} \item L\end{yagitemize*}

```

```

A
  B
    C
      D
        E
          F
            G
              H
                I
                  J
                    K
                      L

```

Avec un tel réglage, comme ce document a un `onerror=nice` au chargement de `yagusylo`, on trouve le texte qui suit dans le fichier `.log` :

```

1 Package yagusylo Warning: Too deeply nested for your setup.
2 (yagusylo) I keep on using the last symbol.
3 (yagusylo) You could have a look at your last
4 (yagusylo) ‘‘setyagitemize’’
5 (yagusylo) First ‘‘yagitemize*’’ too many on input line ***.

```

dans lequel ******* donne le numéro de la ligne sur laquelle se trouve le 5^e `\begin{yagitemize*}` puisque je n'ai donné le réglage explicite que pour seulement quatre niveaux.

9 — yagenumerate et notyagenum

```
11 \begin{yagenumerate}
12   \item Thursday Next;
13   \begin{notyagenum}
14     \begin{enumerate}
15       \item Light armoured brigade;
16       \item SpecOps 27;
17     \end{enumerate}
18   \end{notyagenum}
19   \item Landen Park-Lane;
20 \end{yagenumerate}
```

- ① Thursday Next;
 - (a) Light armoured brigade;
 - (b) SpecOps 27;
- ② Landen Park-Lane;

13 Fichier de configuration

Voici le cœur du fichier de configuration `yagussylo.cfg` fourni, comme exemple, avec cette extension :

```
22 \defyagenumpattern{wastrol}{symfam=wasysym,
23   firstitemnum=88, enumlength=14, symcolor=purple}
```

Quatrième partie

Galerie

14 De l'extension pifont

14.1 Les symboles de la symfam pifont

33 : ✂	34 : ✂	35 : ✂	36 : ✂	37 : 🗑	38 : 🕒
39 : 🕒	40 : ✈	41 : 📧	42 : 🗑	43 : 🗑	44 : 🗑
45 : 📧	46 : 📧	47 : 📧	48 : 📧	49 : 🗑	50 : 🗑
51 : ✓	52 : ✓	53 : ✕	54 : ✕	55 : ✕	56 : ✕
57 : ✚	58 : ✚	59 : ✚	60 : ✚	61 : ✚	62 : ✚
63 : ✚	64 : ✚	65 : ✚	66 : ✚	67 : ✚	68 : ✚
69 : ✚	70 : ✚	71 : ✚	72 : ✚	73 : ✚	74 : ✚
75 : ✚	76 : ✚	77 : ✚	78 : ✚	79 : ✚	80 : ✚
81 : ✚	82 : ✚	83 : ✚	84 : ✚	85 : ✚	86 : ✚
87 : ✚	88 : ✚	89 : ✚	90 : ✚	91 : ✚	92 : ✚
93 : ✚	94 : ✚	95 : ✚	96 : ✚	97 : ✚	98 : ✚
99 : ✚	100 : ✚	101 : ✚	102 : ✚	103 : ✚	104 : ✚
105 : ✚	106 : ✚	107 : ✚	108 : ●	109 : ○	110 : ■
111 : □	112 : □	113 : □	114 : □	115 : ▲	116 : ▼
117 : ◆	118 : ◆	119 : ◐	120 :	121 :	122 :
123 : ◌	124 : ◌	125 : “	126 : ”		
161 : 🍷	162 : ♣	163 : ♠	164 : ♥	165 : ♠	166 : 🍷
167 : 🍷	168 : ♣	169 : ♠	170 : ♥	171 : ♠	172 : ①
173 : ②	174 : ③	175 : ④	176 : ⑤	177 : ⑥	178 : ⑦
179 : ⑧	180 : ⑨	181 : ⑩	182 : ①	183 : ②	184 : ③
185 : ④	186 : ⑤	187 : ⑥	188 : ⑦	189 : ⑧	190 : ⑨
191 : ⑩	192 : ①	193 : ②	194 : ③	195 : ④	196 : ⑤
197 : ⑥	198 : ⑦	199 : ⑧	200 : ⑨	201 : ⑩	202 : ①
203 : ②	204 : ③	205 : ④	206 : ⑤	207 : ⑥	208 : ⑦
209 : ⑧	210 : ⑨	211 : ⑩	212 : →	213 : →	214 : ↔
215 : ↕	216 : ↘	217 : →	218 : ↘	219 : ↘	220 : →
221 : →	222 : →	223 : →	224 : →	225 : →	226 : →
227 : →	228 : →	229 : →	230 : →	231 : →	232 : →
233 : →	234 : →	235 : →	236 : →	237 : →	238 : →
239 : →	240 :	241 : →	242 : →	243 : →	244 : →
245 : →	246 : →	247 : →	248 : →	249 : →	250 : →
251 : →	252 : →	253 : →	254 : →		

15 De l'extension ifsym

15.1 Les symboles de la symfam ifsym

0 : 	1 : 	2 : 	3 : 	4 : 	5 : 
6 : 	7 : 	8 : 	9 : 	10 : 	11 : 
12 : 					
14 : 					
16 : 	17 : 	18 : 	19 : 	20 : 	21 : 
22 : 	23 : 	24 : 	25 : 	26 : 	27 : 
28 : 	29 : 	30 : 	31 : 	32 : 	33 : 
34 : 	35 : 	36 : 	37 : 	38 : 	39 : 
40 : 	41 : 	42 : 			
45 : 	46 : 				
48 : 	49 : 	50 : 	51 : 	52 : 	53 : 
54 : 	55 : 	56 : 	57 : 	58 : 	59 : 
60 : 	61 : 	62 : 	63 : 		
68 : 					
72 : 					
76 : 	77 : 				
100 : 					
108 : 	109 : 				
124 : 					

15.2 Les symboles de la symfam ifsymgeo

0 :	1 :	2 :	3 :	4 :	5 :
6 :	7 :	8 :	9 :		
13 :	14 :	15 :			
26 :	27 :	28 :	29 :	30 :	31 :
32 :	33 :	34 :	35 :	36 :	37 :
38 :					
47 :	48 :	49 :	50 :	51 :	52 :
53 :	54 :				
63 :	64 :	65 :	66 :	67 :	68 :
69 :	70 :				
79 :	80 :	81 :	82 :	83 :	84 :
85 :	86 :				
95 :	96 :	97 :	98 :	99 :	100 :
101 :	102 :				
111 :	112 :	113 :	114 :	115 :	116 :
117 :	118 :				

15.3 Les symboles de la symfam ifsymgeonarrow

0 :	1 :	2 :	3 :	4 :	5 :
6 :	7 :	8 :	9 :		
13 :	14 :	15 :			
26 :	27 :	28 :	29 :	30 :	31 :
32 :	33 :	34 :	35 :	36 :	37 :
38 :					
47 :	48 :	49 :	50 :	51 :	52 :
53 :	54 :				
63 :	64 :	65 :	66 :	67 :	68 :
69 :	70 :				
79 :	80 :	81 :	82 :	83 :	84 :
85 :	86 :				
95 :	96 :	97 :	98 :	99 :	100 :
101 :	102 :				
111 :	112 :	113 :	114 :	115 :	116 :
117 :	118 :				

15.4 Les symboles de la symfam ifsymgeowide

0 :	1 :	2 :	3 :	4 :	5 :
6 :	7 :	8 :	9 :		
13 :	14 :	15 :			
26 :	27 :	28 :	29 :	30 :	31 :
32 :	33 :	34 :	35 :	36 :	37 :
38 :					
47 :	48 :	49 :	50 :	51 :	52 :
53 :	54 :				
63 :	64 :	65 :	66 :	67 :	68 :
69 :	70 :				
79 :	80 :	81 :	82 :	83 :	84 :
85 :	86 :				
95 :	96 :	97 :	98 :	99 :	100 :
101 :	102 :				
111 :	112 :	113 :	114 :	115 :	116 :
117 :	118 :				

15.5 Les symboles de la symfam ifsymweather

0 : ☉	1 : ☉	2 : ☉	3 : ☉	4 : ●	5 : ☉
6 : ☉	7 : ☉	8 : ☉	9 : ☉	10 : ☉	11 : ☉
16 : ☀	17 : ☀	18 : ●	19 : ☁	20 : ☁	21 : ☁
22 : ☁	23 : ☁	24 : ☁	25 : ❄	26 : ☁	27 : ☁
28 : ☁	29 : ☁	30 : ☁	31 : ☁	32 : ☁	33 : ☁
34 : ☁	35 : ☁	36 : ☁			
48 : ☁	49 : ☁	50 : ☁	51 : ☁	52 : ☁	53 : ☁
54 : ☁	55 : ☁	56 : ☁	57 : ☁	58 : ☁	

15.6 Les symboles de la symfam ifsymclock

0 :		1 :		2 :		3 :		4 :		5 :	
6 :		7 :		8 :		9 :		10 :		11 :	
12 :		13 :		14 :		15 :		16 :		17 :	
18 :		19 :		20 :		21 :		22 :		23 :	
24 :		25 :		26 :		27 :		28 :		29 :	
30 :		31 :		32 :		33 :		34 :		35 :	
36 :		37 :		38 :		39 :		40 :		41 :	
42 :		43 :		44 :		45 :		46 :		47 :	
48 :		49 :		50 :		51 :		52 :		53 :	
54 :		55 :		56 :		57 :		58 :		59 :	
60 :		61 :		62 :		63 :		64 :		65 :	
66 :		67 :		68 :		69 :		70 :		71 :	
72 :		73 :		74 :		75 :		76 :		77 :	
78 :		79 :		80 :		81 :		82 :		83 :	
84 :		85 :		86 :		87 :		88 :		89 :	
90 :		91 :		92 :		93 :		94 :		95 :	
96 :		97 :		98 :		99 :		100 :		101 :	
102 :		103 :		104 :		105 :		106 :		107 :	
108 :		109 :		110 :		111 :		112 :		113 :	
114 :		115 :		116 :		117 :		118 :		119 :	
120 :		121 :		122 :		123 :		124 :		125 :	
126 :		127 :		128 :		129 :		130 :		131 :	
132 :		133 :		134 :		135 :		136 :		137 :	
138 :		139 :		140 :		141 :		142 :		143 :	
148 :		149 :		150 :		151 :		152 :		153 :	
154 :		155 :									

16 De l'extension marvosym

16.1 Les symboles de la symfam marvosym

0 : □	1 : □	2 : □	3 : □	4 : □	5 : □
6 : □	7 : □	8 : □	9 : □	10 : □	11 : □
12 : □	13 : □	14 : □	15 : □	16 : □	17 : □
18 : □	19 : □	20 : □	21 : □	22 : □	23 : □
24 : □	25 : □	26 : □	27 : □	28 : □	29 : □
30 : □	31 : □	32 :	33 : ☉	34 : ≡	35 : Δ
36 : Δ	37 : ⚡	38 : ⚡	39 : ⚡	40 : (41 :)
42 : ×	43 : +	44 : ,	45 : -	46 : .	47 : /
48 : 0	49 : 1	50 : 2	51 : 3	52 : 4	53 : 5
54 : 6	55 : 7	56 : 8	57 : 9	58 : →	59 : ⇒
60 : ≤	61 : ≅	62 : ≥	63 : ⇄	64 : @	65 : Ⓞ
66 : ✉	67 : €	68 : €	69 : ⚡	70 : ⚡	71 : ⚡
72 : ⚡	73 : ⚡	74 : ⚡	75 : ⚡	76 : ⚡	77 :
78 : □	79 : □	80 :	81 : ×	82 : --	83 : ×
84 : ⚡	85 : ☉	86 : ⚡	87 : ⚡	88 : ⚡	89 : ☉
90 : ⚡	91 : /	92 : /	93 : ≡	94 : ≠	95 : /
96 : ⚡	97 : ⚡	98 : ⚡	99 : €	100 : €	101 : €
102 : ⚡	103 : ⚡	104 : ⚡	105 : ⚡	106 : ⚡	107 : ⚡
108 : ↓	109 : ⚡	110 : ⚡	111 : ⚡	112 : ⚡	113 : ×
114 : --	115 : ×	116 : ⚡	117 : FAX	118 : ⚡	119 : ⚡
120 : ⚡	121 : ⚡	122 : ⚡	123 : ○	124 : ♂	125 : ♀
126 : ♀	127 : ♂	128 : ♀	129 : ♀	130 : □	131 : ♂
132 : ♀	133 : ♂	134 : †	135 : †	136 : †	137 : ☐
138 : ☐	139 : ☐	140 : ♥	141 : @	142 : ⚡	143 : □
144 : ☐	145 : ●	146 : ●	147 : ■	148 : ■	149 : ●
150 : -	151 : □	152 : □	153 : L	154 : I	155 : ○
156 : T	157 : L	158 : I	159 : T	160 : ϕ	161 : β
162 : ⚡	163 : ⚡	164 : €	165 : ⚡	166 : \$	167 : ☉
168 : ☉	169 : ☉	170 : ☉	171 : ☉	172 : ☉	173 : ☉
174 : ⚡	175 : ⚡	176 : ⚡	177 : ⚡	178 : ⚡	179 : □
180 : ⏪	181 : ⏩	182 : ◀	183 : ▶	184 : ▶	185 : ▶
186 : ▲	187 : ▼	188 : ▲	189 : ▼	190 : ☉	191 : ☉
192 : ☉	193 : ☉	194 : ♀	195 : ♀	196 : ♂	197 : 4
198 : ⚡	199 : ⚡	200 : ⚡	201 : ♀	202 : ♂	203 : Δ
204 : ⚡	205 : ⚡	206 : ⚡	207 : ⚡	208 : ⚡	209 : ⚡
210 : ⚡	211 : ⚡	212 : ⚡	213 : ⚡	214 : ⚡	215 : ⚡
216 : ⚡	217 : ⚡	218 : ⚡	219 : ⚡	220 : ⚡	221 : ⚡
222 : □	223 : □	224 : ♀	225 : ♂	226 : II	227 : ⚡
228 : ⚡	229 : ⚡	230 : Ω	231 : ⚡	232 : /	233 : ⚡
234 : ⚡	235 : ⚡	236 : □	237 : □	238 : □	239 : □
240 : A	241 : p	242 : □	243 : □	244 : □	245 : □
246 : □	247 : ·	248 : □	249 : □	250 : □	251 : □
252 : □	253 : ⚡	254 : ⚡	255 : ⚡		

17 De l'extension fourier

17.1 Les symboles de la symfam fourier

65 : ☹

66 : ⚠

69 : €

76 : 🍷

77 : ☹

78 : 🍷

84 : ✖

85 : ✖

88 : ✖

89 : 🍷

90 : 🍷

91 : 🍷

92 : 🍷

93 : ✖

102 : 🍷

103 : 🍷

104 : 🍷

106 : 🍷

109 : 🍷

110 : 🍷

111 : 🍷

116 : 🍷

117 : 🍷

18 De l'extension wasysym

18.1 Les symboles de la symfam wasysym

0 : Δ	1 : \triangleleft	2 : \trianglelefteq	3 : \triangleright	4 : \trianglerighteq	5 : \therefore
6 : \circ	7 : ☎	8 : \checkmark	9 : \clubsuit	10 : \spadesuit	11 : \blacktriangleright
12 : \downarrow	13 : \downarrow	14 : \circ	15 : \blacktriangleright	16 : \blacktriangleleft	17 : \blacktriangleright
18 : ζ	19 : Ω	20 : \cup	21 : \otimes	22 : \oplus	23 : Υ
24 : \neg	25 : ♀	26 : ♂	27 : \otimes	28 : \oplus	29 : \propto
30 : \sphericalangle	31 : ∅	32 : \bullet	33 : \circlearrowright	34 : \circlearrowleft	35 : \circ
36 : \complement	37 : D	38 : ♁	39 : ♀	40 : \prec	41 : \succ
42 : $\hat{}$	43 : \checkmark	44 : \odot	45 : \odot	46 : \otimes	47 : \odot
48 : \cup	49 : \otimes	50 : \square	51 : \diamond	52 : \boxtimes	53 : \boxplus
54 : \clubsuit	55 : \circ	56 : \bigcirc	57 : \circ	58 : \sim	59 : \rightsquigarrow
60 : \square	61 : \square	62 : \lesssim	63 : \gtrsim	64 : \approx	65 : \ast
66 : \ast	67 : \ast	68 : \diamond	69 : \ast	70 : ∇	71 : \blacktriangleleft
72 : \blacktriangleright	73 : \triangleleft	74 : D	75 : \blacktriangle	76 : \blacktriangledown	
80 : γ	81 : \curvearrowright	82 : \curvearrowleft			
85 : ⊖	86 : ♂	87 : ♂	88 : ♁	89 : ♁	90 : ♁
91 : ♁	92 : ♁	93 : ♁	94 : ♁	95 : ♁	96 : ♁
97 : ♁	98 : ♁	99 : ♁	100 : ♁	101 : ♁	102 : ♁
103 : ♁	104 : ‰	105 : ♁	106 : ♁	107 : ♁	108 : ♁
109 : ♁	110 : ♁	111 : ♁	112 : ♁	113 : ♁	114 : \int
115 : \iint	116 : \iiint	117 : ♁	118 : ♁	119 : \int	120 : \int
121 : \iiint	122 : ♁	123 : ♁	124 : ♁	125 : ♁	126 : ♁
127 : ♁					

19 De l'extension bbding

19.1 Les symboles de la symfam bbding

0 : 	1 : 	2 : 	3 : 	4 : 	5 : 
6 : 	7 : 	8 : 	9 : 	10 : 	11 : 
12 : 	13 : 	14 : 	15 : 	16 : 	17 : 
18 : 	19 : 	20 : 	21 : 	22 : 	23 : 
24 : 	25 : 	26 : 	27 : 	28 : 	29 : 
30 : 	31 : 	32 : 	33 : 	34 : 	35 : 
36 : 	37 : 	38 : 	39 : 	40 : 	41 : 
42 : 	43 : 	44 : 	45 : 	46 : 	47 : 
48 : 	49 : 	50 : 	51 : 	52 : 	53 : 
54 : 	55 : 	56 : 	57 : 	58 : 	59 : 
60 : 	61 : 	62 : 	63 : 	64 : 	65 : 
66 : 	67 : 	68 : 	69 : 	70 : 	71 : 
72 : 	73 : 	74 : 	75 : 	76 : 	77 : 
78 : 	79 : 	80 : 	81 : 	82 : 	83 : 
84 : 	85 : 	86 : 	87 : 	88 : 	89 : 
90 : 	91 : 	92 : 	93 : 	94 : 	95 : 
96 : 	97 : 	98 : 	99 : 	100 : 	101 : 
102 : 	103 : 	104 : 	105 : 	106 : 	107 : 
108 : 	109 : 	110 : 	111 : 	112 : 	113 : 
114 : 	115 : 	116 : 	117 : 	118 : 	119 : 
120 : 	121 : 	122 : 	123 : 		

20 De l'extension dingbat

20.1 Les symboles de la symfam dingbat

66 : 

67 : 

68 : 

73 : 

78 : 

79 : 

83 : 

90 : 

97 : 

98 : 

99 : 

100 : 

101 : 

102 : 

103 : 

104 : 

20.2 Symboles larges de dingbat



20.3 Les symboles de la symfam ark

67 :  68 :  69 : 

76 : 

80 : 

82 : 

85 : 

87 : 

100 : 

117 : 

Index

- [*](#), [5–7](#), [10](#), [12](#), [14](#)
- [after \(CLEF\)](#), [8](#)
- [before \(CLEF\)](#), [8](#)
- [boxwidth \(CLEF\)](#), [8](#)
- CLÉ
 - [after](#), [8](#)
 - [before](#), [8](#)
 - [boxwidth](#), [8](#)
 - [color](#), [12](#)
 - [configfile](#), [12](#)
 - [enumlength](#), [11](#)
 - [enumpattern](#), [11](#)
 - [firstitemnum](#), [11](#)
 - [head](#), [9](#)
 - [leadtype](#), [8](#)
 - [symcolor](#), [6](#)
 - [symcolor \[enum\]](#), [11](#)
 - [symfam](#), [6](#)
 - [symfam \[enum\]](#), [11](#)
 - [symplace](#), [8](#)
 - [sympos](#), [8](#)
 - [tail](#), [9](#)
 - [XcolorOptions](#), [12](#)
- [color \(CLEF\)](#), [12](#)
- [configfile \(CLEF\)](#), [12](#)
- [\defdingname](#), [6](#)
- [\defdingname+](#), [7](#)
- [enumlength \(CLEF\)](#), [11](#)
- [enumpattern \(CLEF\)](#), [11](#)
- environnement
 - [yagenumerate](#), [12](#)
 - [yagitemize](#), [10](#)
 - [yagitemize*](#), [10](#)
- [firstitemnum \(CLEF\)](#), [11](#)
- [head \(CLEF\)](#), [9](#)
- [leadtype \(CLEF\)](#), [8](#)
- [\newenumpattern](#), [11](#)
- PACKAGE
 - [bding](#), [31](#)
 - [dingbat](#), [32](#)
 - [fourier](#), [29](#)
 - [ifsym](#), [22](#)
 - [marvosym](#), [28](#)
 - [pifont](#), [21](#)
 - [wasysym](#), [30](#)
- [\setyagenumeratekeys](#), [12](#)
- [\setyagitemize](#), [10](#)
- [\setyagitemize*](#), [10](#)
- [\setyagusylokeys](#), [6](#)
- [symcolor \(CLEF\)](#), [6](#)
- [symcolor \[enum\] \(CLEF\)](#), [11](#)
- SYMFAM
 - [ark](#), [34](#)
 - [bding](#), [31](#)
 - [dingbat](#), [32](#)
 - [fourier](#), [29](#)
 - [ifsym](#), [22](#)
 - [ifsymclock](#), [27](#)
 - [ifsymgeo](#), [23](#)
 - [ifsymgeonarrow](#), [24](#)
 - [ifsymgeowide](#), [25](#)
 - [ifsymweather](#), [26](#)
 - [marvosym](#), [28](#)
 - [pifont](#), [21](#)
 - [wasysym](#), [30](#)
- [symfam \(CLEF\)](#), [6](#)
- [symfam \[enum\] \(CLEF\)](#), [11](#)
- [symplace \(CLEF\)](#), [8](#)
- [sympos \(CLEF\)](#), [8](#)
- [tail \(CLEF\)](#), [9](#)
- [XcolorOptions \(CLEF\)](#), [12](#)
- [\yagding](#), [6](#)
- [\yagding*](#), [7](#)
- [\yagding+](#), [7](#)
- [yagenumerate \(environnement\)](#), [12](#)
- [\yagfill](#), [8](#)
- [\yagfill*](#), [8](#)
- [\yagfill+](#), [8](#)
- [yagitemize \(environnement\)](#), [10](#)
- [yagitemize* \(environnement\)](#), [10](#)
- [\yagline](#), [9](#)
- [\yagline*](#), [9](#)
- [\yagline+](#), [9](#)