Démonstration de l'utilisation du package"algorithme 1.1"

Pierre Chatelier
e-mail : pierre.chatelier@club-internet.fr
http ://perso.club-internet.fr/ktd

17 décembre 2004

Table des matières

La table des algorithmes est insérée ci-dessous par la commande :

```
\listofalgorithms
```

Liste des Algorithmes

1 Localisation automatique des mots-clés

Selon que

`\usepackage[french]{algorithme}`

ou

`\usepackage[english]{algorithme}`

ait été employé, les mots-clés seront dans la langue choisie. La langue est héritée du package `babel` si elle n'est pas spécifiée, et il s'agit du par défaut du Français.

2 Environnement algorithme

Pour écrire un algorithme, on ouvre l'environnement adéquat. Il est automatiquement encapsulé dans une boîte.

```
\begin{algorithme}
Je suis un algorithme\\
\end{algorithme}
```

Je suis un algorithme

3 Légende : \caption

Pour ajouter une légende, et voir l'algorithme apparaître dans la table des algorithmes, on utilise le *caption*. La légende est inscrite en dessous.

```
\begin{algorithme}
Encore un algorithme\\
\caption{J'aime les légendes}
\end{algorithme}
```

Encore un algorithme

Algorithme 1: J'aime les légendes

4 Typage : \Type

On peut préciser le type des variables.

```
\begin{algorithme}
\Type{x}{bool}; \\
\Type{y}{integer}; \\
x $\leftarrow$ \True; \\
y $\leftarrow$ 0;
\caption{Typage}
\end{algorithme}
```

x : **bool** ;
y : **integer** ;
x ← **vrai** ;
y ← 0 ;

Algorithme 2: Typage

5 Commentaires : \Rem

On peut ajouter des commentaires.

```
\begin{algorithme}
\Rem{$x$ est initialisé à 0}\\
x $\leftarrow$ 0;
\caption{Commentaires}
\end{algorithme}
```

[x est initialisé à 0]
x ← 0 ;

Algorithme 3: Commentaires

6 Boucle tant que : \While ou \WhileDo

Boucle "Tant que"

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 1; \\
\While{x $\leq$ 5}
    {x $\leftarrow$ x+1;}
\caption{Tant que}
\end{algorithme}
```

x : **entier** ;
x ← 1 ;
Tant que (x ≤ 5) **faire**
  | x ← x+1 ;
Fait

Algorithme 4: Tant que

7 Boucle "Tant que (version courte)" : \While* ou \WhileDo*

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 0; \\
\While*{x $\leq$ 5}
{x $\leftarrow$ x+1}
\caption{Tant que
(version courte)}
\end{algorithme}
```

x : **entier** ;
x ← 0 ;
Tant que (x ≤ 5) **faire** x ← x+1 **Fait**

Algorithme 5: Tant que (version courte)

Pratiquement toutes les structures de contrôle de ce package ont une forme courte (forme étoilée dans le fichier source LaTeX 2ε) (Une exception : le Selon que).

8 Boucle "Répéter. . .jusqu'à" : \Repeat ou \RepeatUntil

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 0; \\
\Repeat{x $\leftarrow$ x+1;}
{x = 6}
\caption{Répéter \ldots jusqu'à}
\end{algorithme}
```

x : **entier** ;
x ← 0 ;
Répéter
  | x ← x+1 ;
jusqu'à ce que (x = 6)

Algorithme 6: Répéter . . .jusqu'à

9 Boucle “Pour” standard : \ForFromTo ou \ForFromToDo

```
\begin{algorithme}
\Type{x}{entier}; \\
\ForFromTo{x}{0}{5}
    {x $\leftarrow$ x+1}
\caption{boucle Pour standard}
\end{algorithme}
```

x : **entier** ;
Pour x **de** 0 **à** 5 **faire**
  | x ← x+1
Fin Pour

Algorithme 7: boucle Pour standard

10 Boucle “Pour” de style C : \For ou \ForDo

```
\begin{algorithme}
\Type{x}{entier}; \\
\For{(x=0 ; x $\leq$ 5; x++)}
    {\Rem{Rien à faire, du coup}}
\caption{boucle Pour en style C}
\end{algorithme}
```

x : **entier** ;
Pour (x=0 ; x ≤ 5 ; x++) **faire**
  | *[Rien à faire, du coup]*
Fin Pour

Algorithme 8: boucle Pour en style C

11 Si...Alors : \If ou \IfThen

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 0; \\
\If{x = 0}{x $\leftarrow$ 1;}
\caption{Si \ldots alors \ldots}
\end{algorithme}
```

x : **entier** ;
x ← 0 ;
Si (x = 0) **Alors**
  | x ← 1 ;
Fin Si

Algorithme 9: Si ...alors ...

12 Si...Alors...Sinon : \IfThenElse

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 0; \\
\IfThenElse{x = 0}
    {x $\leftarrow$ 1;}
    {x $\leftarrow$ 2;}
\caption{Si \ldots alors
\ldots sinon}
\end{algorithme}
```

x : **entier** ;
x ← 0 ;
Si (x = 0) **Alors**
  | x ← 1 ;
Sinon
  | x ← 2 ;
Fin Si

Algorithme 10: Si ...alors ...sinon

13 Quelques mots-clés : \True, \False, \Or, \And, \Not

```
\begin{algorithme}
\Type{x}{bool}; \\
\If{condition1 \And
    (Condition2 \Or
    \Not(condition3))}
    {x $\leftarrow$ \False}
\caption{Utilisation de quelques
mots-clefs}
\end{algorithme}
```

x : **bool** ;
Si (condition1 **ET** (Condition2 **OU** ¬(condition3))) **Alors**
  | x ← **faux**
Fin Si

Algorithme 11: Utilisation de quelques mots-clefs

14 Structure “Selon que” : \Switch

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 0; \\
\Switch{
    x = 0 : x $\leftarrow$ 1;\\
    x = 1 : x $\leftarrow$ 11;\\
    x = 2 : x $\leftarrow$ 111;
}
\caption{Selon que}
\end{algorithme}
```

x : **entier** ;
x ← 0 ;
Selon que
  | x = 0 : x ← 1 ;
  | x = 1 : x ← 11 ;
  | x = 2 : x ← 111 ;
Fin Selon que

Algorithme 12: Selon que

14.1 Selon que raté

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 0; \\
\Type{y}{entier}; \\
y $\leftarrow$ 0; \\
\Switch{
    x = 0 : x $\leftarrow$ 1;\\
            y $\leftarrow$ 2;\\
    x = 1 : x $\leftarrow$ 11;\\
            y $\leftarrow$ 22;\\
    x = 2 : x $\leftarrow$ 111;\\
            y $\leftarrow$ 222;\\
}
\caption{Selon que raté}
\end{algorithme}
```

x : **entier** ;
x ← 0 ;
y : **entier** ;
y ← 0 ;
Selon que
  | x = 0 : x ← 1 ;
  | y ← 2 ;
  | x = 1 : x ← 11 ;
  | y ← 22 ;
  | x = 2 : x ← 111 ;
  | y ← 222 ;
Fin Selon que

Algorithme 13: Selon que raté

14.2 Selon que corrigé

```
\begin{algorithme}
\Type{x}{entier}; \\
x $\leftarrow$ 0; \\
\Type{y}{entier}; \\
y $\leftarrow$ 0; \\
\Switch{
  x=0:\Block{
        x $\leftarrow$ 1;\\
        y $\leftarrow$ 2;\\
       }\\
  x=1:\Block{
        x $\leftarrow$ 11;\\
        y $\leftarrow$ 22;\\
       }\\
  x=2:\Block{
        x $\leftarrow$ 111;\\
        y $\leftarrow$ 222;\\
       }
}
\caption{Selon que corrigé}
\end{algorithme}
```

x : **entier** ;
x ← 0 ;
y : **entier** ;
y ← 0 ;
Selon que
  x = 0 : x ← 1 ;
          y ← 2 ;
  x = 1 : x ← 11 ;
          y ← 22 ;
  x = 2 : x ← 111 ;
          y ← 222 ;
Fin Selon que

Algorithme 14: Selon que corrigé

15 Fonction : \Function

```
\begin{algorithme}
\Function{maFonction}
    {x: \textbf{entier}}{entier}
{
    x $\leftarrow$ x*x*x;\\
    \Return x;
}
\caption{fonction}
\end{algorithme}
```

Fonction maFonction(x : **entier**) : **entier**
  x ← x*x*x ;
  Retourner x ;
Fin

Algorithme 15: fonction

16 Procédure

```
\begin{algorithme}
\Procedure{maProcedure}{x}
{
    x $\leftarrow$ 0;
}
\caption{procédure}
\end{algorithme}
```

Procédure maProcedure(x)
  x ← 0 ;
Fin

Algorithme 16: procédure

17 Imbrication des commandes

L'intérêt de ce package est bien sûr de pouvoir imbriquer les commandes.

```
\begin{algorithme}
\Function{max}
    {\Type{x, y, z}{entier}}
    {entier}
{
  \Type{result}{entier}\\
  \IfThenElse{x $\leq$ y}
  {
    \IfThenElse{y $\leq$ z}
      {result $\leftarrow$ z;}
      {result $\leftarrow$ y;}
  }
  {
    \IfThenElse{x $\leq$ z}
      {result $\leftarrow$ z;}
      {result $\leftarrow$ x;}
  }\\
  \Return{result};
}
\caption{Un algorithme complet}
\end{algorithme}
```

Fonction max(x, y, z : **entier**) : **entier**
  result : **entier**
  Si (x ≤ y) **Alors**
    Si (y ≤ z) **Alors**
      result ← z ;
    Sinon
      result ← y ;
    Fin Si
  Sinon
    Si (x ≤ z) **Alors**
      result ← z ;
    Sinon
      result ← x ;
    Fin Si
  Fin Si
  Retourner result ;
Fin

Algorithme 17: Un algorithme complet