

# sim-os-menus [fr]

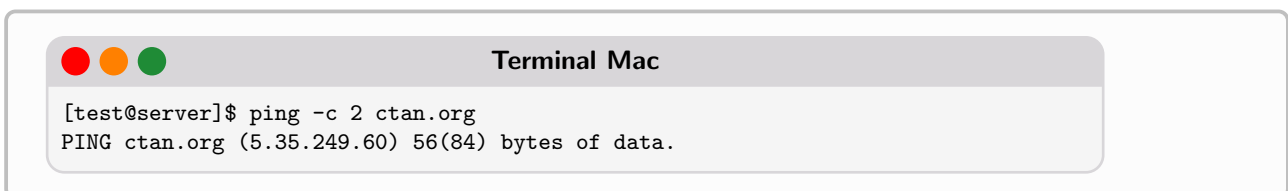
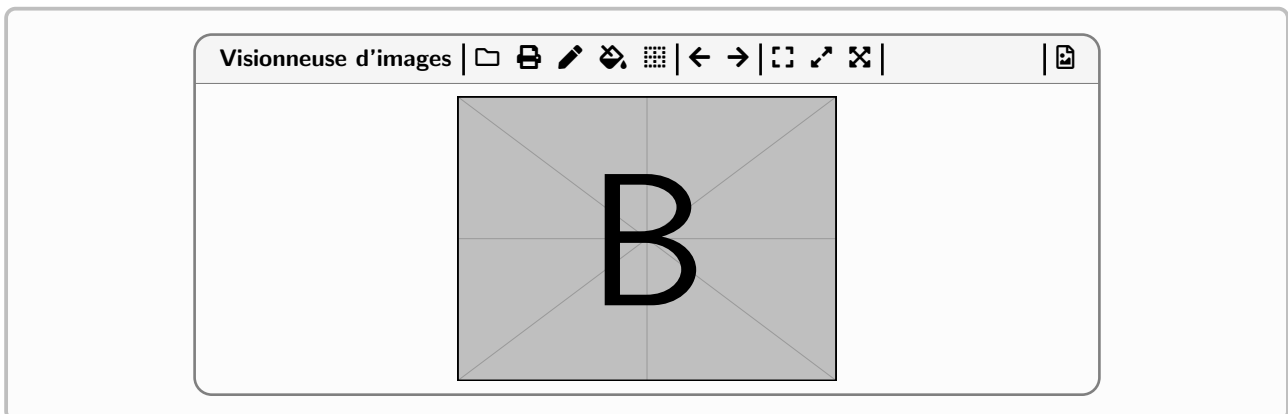
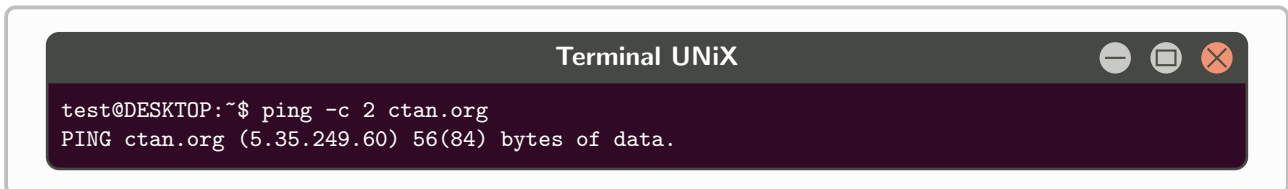
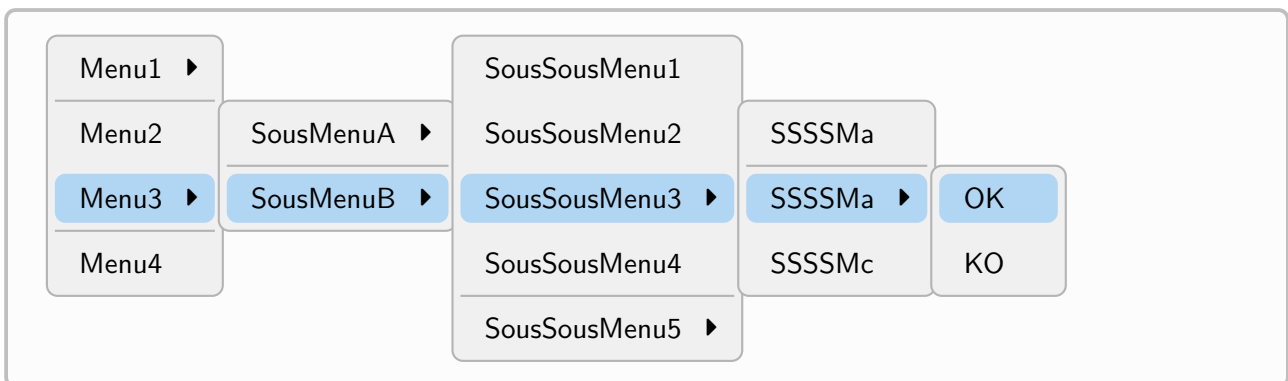
Présenter un 'terminal', une 'visionneuse' ou un 'menu contextuel' comme sur un ordi.

Version 0.1.3 -- 28/09/2024

Cédric Pierquet

c pierquet -- at -- outlook . fr

<https://github.com/cpierquet/sim-os-menus>



# Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Description	2
1.2	Chargement	2
1.3	Historique	2
<b>2</b>	<b>Les commandes disponibles</b>	<b>3</b>
2.1	Menu contextuel	3
2.2	Terminal	5
2.3	Visionneuses	7
2.4	Arborescence de dossiers/fichiers	10

---

## 1 Introduction

### 1.1 Description

Avec ce packages, il est possible de créer un « faux » terminal, ou un « faux » menu contextuel, ou encore une « fausse » visionneuse, comme sur un écran d'ordinateur !

Les styles sont *globalement* fixés, mais quelques personnalisations sont possibles.

### 1.2 Chargement

Pour charger le package, il suffit d'ajouter dans le préambule :

```
\usepackage{sim-os-menus}
```

Les packages utiles, et chargés sont :

- `tikz` (avec `calc`, `positioning`), `pgf`, `pgffor`;
- `calc`, `fontawesome5`, `forest` (avec `edges`);
- `simplekv`, `xintexpr`, `listofitems`, `xstring`;
- `settobox`, `tabularray`;
- `tcolorbox` (avec `breakable`, `fitting`, `skins`, `listings`, `listingsutf8`, `hooks`).

### 1.3 Historique

- 0.1.2 : Arborescence type explorateur
- 0.1.2 : Visionneuse 'like' de scripts
- 0.1.1 : Ajout des commandes en [fr]
- 0.1.0 : Version initiale

## 2 Les commandes disponibles

### 2.1 Menu contextuel

Dans le but de créer un menu contextuel, la commande est :

```
%----menu contextuel
\MenuContextuel[clés]{liste des items}<options tikz>
```

Les clés, optionnelles et entre [...], sont :

- `CoulFond` := couleur du fond;
- `CoulSurlig` := couleur du surlignage;
- `Arrondi` := booléen pour arrondir les coins (`true` par défaut);
- `Police` := police pour les items (`\normalsize\normalfont` par défaut);
- `CoulItems` := couleur(s) des items (`black` par défaut);
- `MargeV` := marge verticale des lignes (`6pt` par défaut);
- `MargeH` := marge horizontale des lignes (`12pt` par défaut);
- `Fleche` := caractère pour la flèche (`\faCaretRight` par défaut);
- `ListeSeps` := liste des éventuels séparateurs (vide ou pour tous les niveaux!);
- `ListeIcones` := liste des éventuelles icônes (vide ou pour tous les niveaux/items!);
- `ListeDecals` := liste des éventuels décalages verticaux pour les sous-menus (à partir du n°2, ...!) (vide ou pour tous les sous-niveaux!);
- `Icones` := booléen pour afficher les icônes (`false` par défaut);
- `Barre` := booléen pour une petite barre verticale après les éventuelles icônes `small vertical bar with icons` (`true` par défaut);
- `Space` := espacement horizontal entre les sous-menus (`-0.125` par défaut).

L'argument obligatoire, entre {...}, correspond à la liste des items, sous la forme :

```
item1A,item1B,... § item2A,item2B,... § ...
```

- si un item se termine par (\*), cela correspond au *début* du sous-menu suivant (un seul par niveau!);
- si un item se termine par (>) (avant un éventuel (\*)), une *flèche* est écrite à la fin de la ligne de l'item.

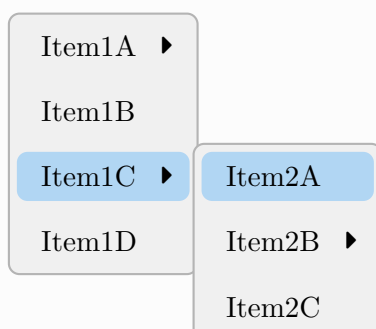
Une syntaxe correcte est fondamentale pour que le code présente correctement le menu contextuel!

Quelques conseils, du fait des clés sensibles `ListeIcones`, `ListeDecals` et `ListeSeps` :

- `ListeIcones` doit avoir le même nombre d'éléments que le nombre de menus/items (avec des items vides possibles);
- `ListeSeps` doit avoir le même nombre d'éléments que le nombre de niveaux (avec des niveaux vides possibles);
- `ListeDecals` doit avoir le même nombre d'éléments que le nombre de sous-menus (Avec 0 si pas de décalage!).

*%style par défaut*

```
\MenuContextuel{Item1A(>),Item1B,Item1C(>)(*),Item1D § Item2A(*),Item2B(>),Item2C}
```

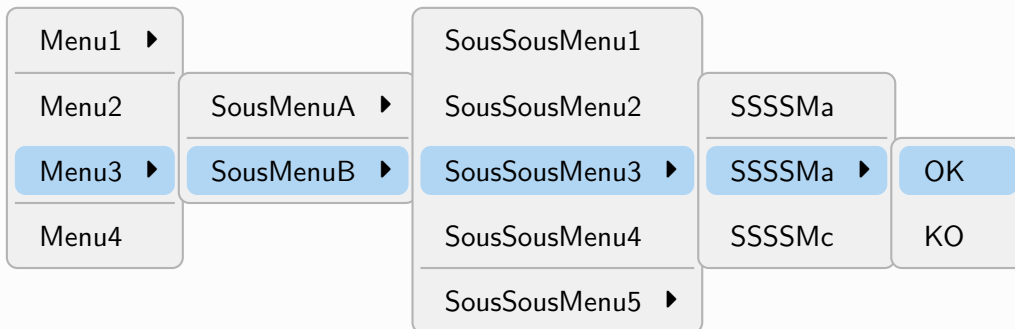


*%style personnalis *

```
\MenuContextuel[Arrondi=false,CoulFond=yellow,CoulSurlign=teal,%  
ListeSeps={1,3/2},CoulItems={black/white},Icones,Espacement=0,%  
ListeIcones={\faCut,\faIcon[regular]{copy},\faCopy,\faPaste / \faNotEqual}]  
{Item1A(>)(*),Item1B,Item1C(>),Item1D § Item2A(*),Item2B(>),Item2C}
```

✂ Item1A ▶	≠ Item2A
📄 Item1B	Item2B ▶
📄 Item1C ▶	Item2C
📄 Item1D	

```
\MenuContextuel[Police=\sffamily,ListeSeps={1,3/1/4/1/},ListeDecals={1,2,1,0}]{%  
Menu1(>),Menu2,Menu3(>)(*),Menu4 §  
SousMenuA(>),SousMenuB(>)(* ) §  
SousSousMenu1,SousSousMenu2,SousSousMenu3(>)(*),SousSousMenu4,SousSousMenu5(> ) §  
SSSSMa,SSSSMa(>)(*),SSSSMc §  
OK(*),KO  
}
```



## 2.2 Terminal

Pour créer un « faux » terminal (Win/UNiX/Mac), les environnements sont :

```
%----Terminal Windows
\begin{TerminalWin}[clés]{options tcolorbox}
...
\end{TerminalWin}

%----Terminal UNiX
\begin{TerminalUnix}[clés]{options tcolorbox}
...
\end{TerminalUnix}

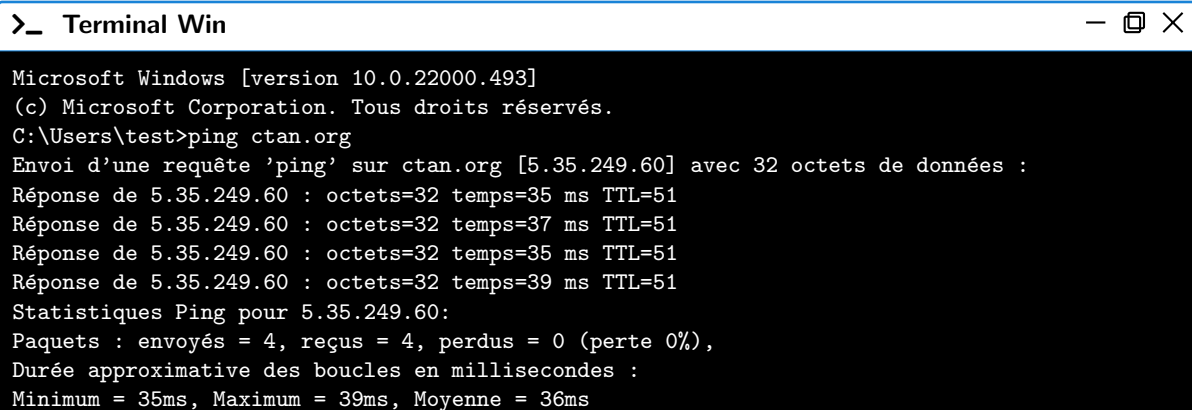
%----Terminal OSX
\begin{TerminalMac}[clés]{options tcolorbox}
...
\end{TerminalMac}
```

Les clés, optionnelles et entre [...], sont :

- **Titre** := titre du terminal (Terminal Win/UNiX/Mac par défaut) ;
- **Align** := alignement horizontal (paragraphe) du terminal (**center** par défaut) ;
- **Largeur** := largeur du terminal (`\linewidth` par défaut) ;
- **Icones** := booléen pour afficher des icones (**true** par défaut).

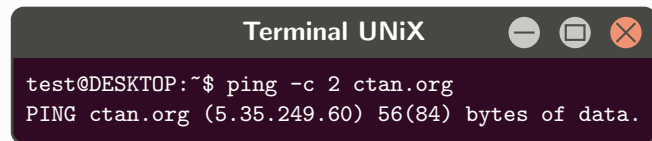
L'argument obligatoire, entre {...}, correspond à des options spécifiques, à passer à la `tcolorbox`.

```
\begin{TerminalWin}{}
Microsoft Windows [version 10.0.22000.493]
(c) Microsoft Corporation. Tous droits réservés.
C:\Users\test>ping ctan.org
Envoi d'une requête 'ping' sur ctan.org [5.35.249.60] avec 32 octets de données :
Réponse de 5.35.249.60 : octets=32 temps=35 ms TTL=51
Réponse de 5.35.249.60 : octets=32 temps=37 ms TTL=51
Réponse de 5.35.249.60 : octets=32 temps=35 ms TTL=51
Réponse de 5.35.249.60 : octets=32 temps=39 ms TTL=51
Statistiques Ping pour 5.35.249.60:
Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
Minimum = 35ms, Maximum = 39ms, Moyenne = 36ms
\end{TerminalWin}
```

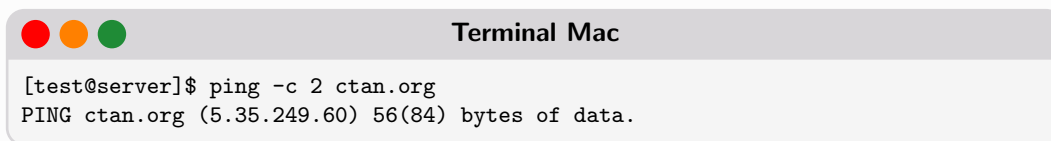


The screenshot shows a terminal window with a title bar that says "Terminal Win" and standard window controls (minimize, maximize, close). The terminal content is identical to the code block above, displaying the output of a ping command to ctan.org.

```
\begin{TerminalUnix}[Align=flush right]{hbox}
test@DESKTOP:~$ ping -c 2 ctan.org
PING ctan.org (5.35.249.60) 56(84) bytes of data.
\end{TerminalUnix}
```



```
\begin{TerminalMac}[Largeur=14cm,Align=flush left]{}
[test@server]$ ping -c 2 ctan.org
PING ctan.org (5.35.249.60) 56(84) bytes of data.
\end{TerminalMac}
```



## 2.3 Visionneuses

Pour créer une « fausse » visionneuse, les environnements sont :

```
%----Visionneuse PDF
\begin{VisionnPDF}[clés]{options tcolorbox}
    ....
\end{VisionnPDF}

%----Visionneuse d'images
\begin{VisionnIMG}[clés]{options tcolorbox}
    ....
\end{VisionnIMG}

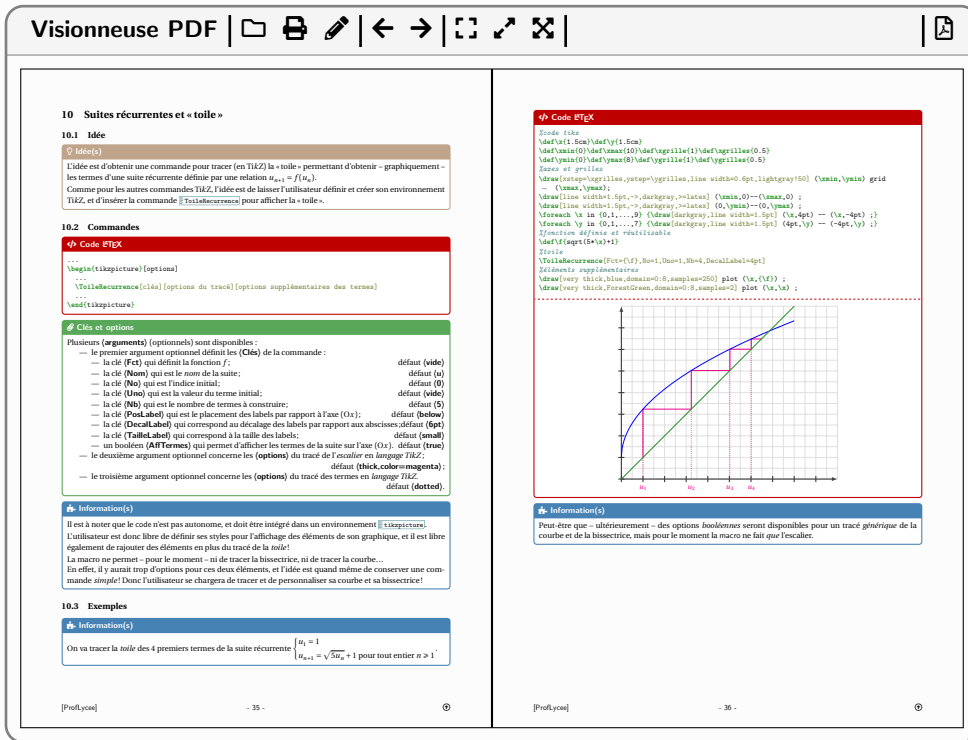
%----Visionneuse de script python
\begin{VisionnPY}[clés]{options tcolorbox}
    ....
\end{VisionnPY}
```

Les clés, optionnelles et entre [...], sont :

- **Titre** := titre de la visionneuse par défaut ;
- **Align** := alignement horizontal (paragraphe) de la visionneuse (**center** par défaut) ;
- **AlignH** := alignement horizontal (du contenu) de la visionneuse (**center** par défaut) ;
- **Largeur** := largeur de la visionneuse (\linewidth par défaut) ;
- **Icones** := booléen pour afficher des icones (**true** par défaut).

L'argument obligatoire, entre {...}, correspond à des options spécifiques, à passer à la tcolorbox.

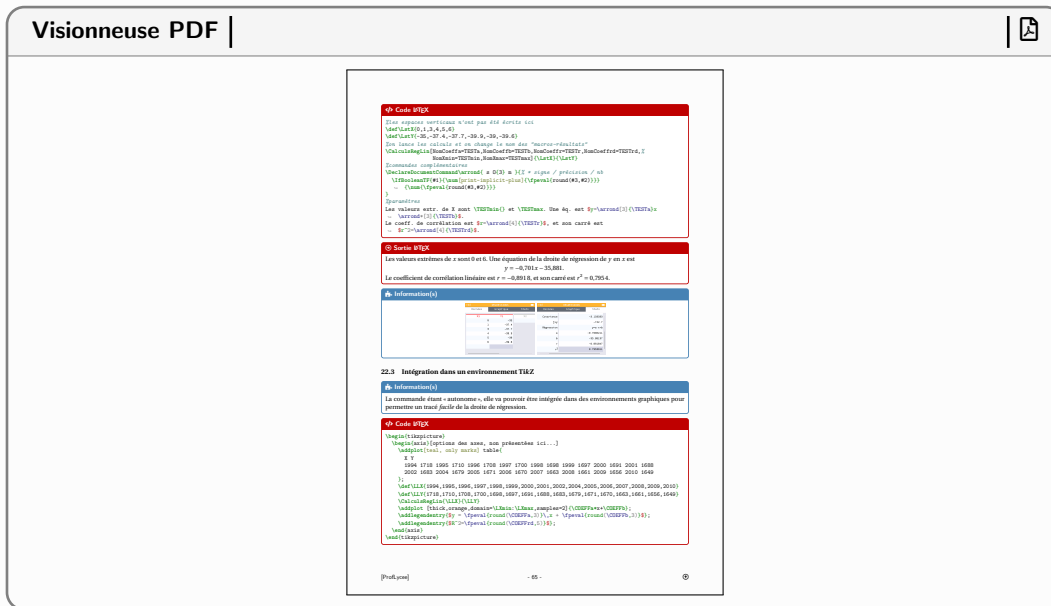
```
\begin{VisionnPDF}{hbox}
\fbbox{\includegraphics[page=35,width=6cm]{ProfLycee-doc.pdf}}%
\fbbox{\includegraphics[page=36,width=6cm]{ProfLycee-doc.pdf}}
\end{VisionnPDF}
```



```

\begin{VisionnPDF}[Largeur=14cm,Icones=false]{}
\fbbox{\includegraphics[page=65,width=4.75cm]{ProfLycee-doc.pdf}}
\end{VisionnPDF}

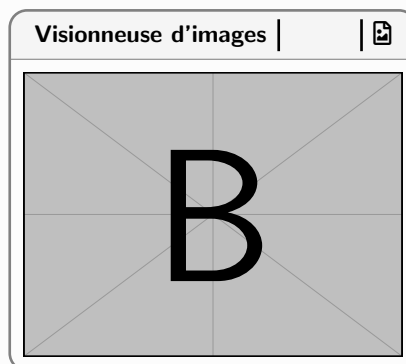
```



```

\begin{VisionnIMG}[Icones=false]{hbox}
\includegraphics[width=5cm]{example-image-b}
\end{VisionnIMG}

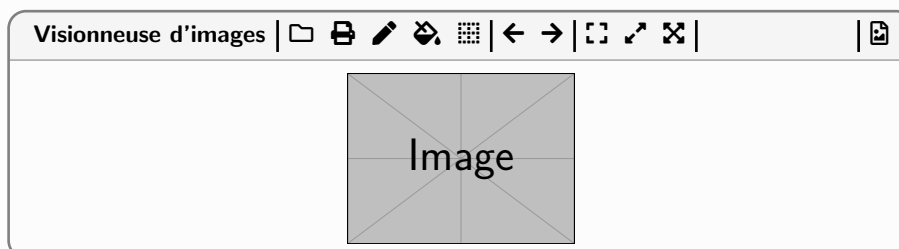
```



```

\begin{VisionnIMG}[Largeur=12cm]{}
\includegraphics[width=3cm]{example-image}
\end{VisionnIMG}

```





```

%avec un environnement de présentation de codes si nécessaire
\begin{VisionnPY}[Largeur=12cm]{ }
\begin{lstlisting}%
[
    language=python,basicstyle=\ttfamily\small,
    keywordstyle=\color{green!50!black},tabsize=4,
    keywordstyle={ [2] \color{magenta}},
    numbers=left,numbersep=3mm,xleftmargin=5mm,
    aboveskip=0pt,belowskip=0pt,
    numberstyle=\footnotesize\ttfamily\color{gray}
]
nterms = int(input("Entrez un nombre: "))

n1 = 0
n2 = 1

print("\n la suite Fibonacci est : ")
print(n1, ", ", n2, end=", ")

for i in range(2, nterms):
    suivant = n1 + n2
    print(suivant, end=", ")

n1 = n2
n2 = suivant
\end{lstlisting}
\end{VisionnPY}

```

```

Éditeur Python | [Icons] | [Run] | [Stop] | [Refresh] | [Close]
1  nterms = int(input("Entrez un nombre: "))
2
3  n1 = 0
4  n2 = 1
5
6  print("\n la suite Fibonacci est : ")
7  print(n1, ", ", n2, end=", ")
8
9  for i in range(2, nterms):
10     suivant = n1 + n2
11     print(suivant, end=", ")
12
13  n1 = n2
14  n2 = suivant

```

## 2.4 Arborescence de dossiers/fichiers

Le package `forest`, et sa librairie `edges`, permet de présenter des arbres, à la manière d'une arborescence de fichiers.

```
\begin{ArborDossiers}[clés]{options forest}
  [dossier,ADdos
    [sous-dossier,ADdos
      [sous-dossier,ADdos
        [fichier1,ADfic]
        [fichier2,ADfic]
      ]
    ]
  ]
\end{ArborDossiers}
```

Les clés, optionnelles et entre [...], sont :

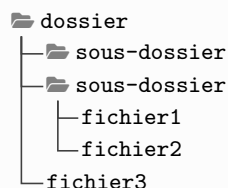
- `Police` := police des labels;
- `CoulIcoD` := couleur des icônes dossier (`gray` par défaut);
- `CoulIcoF` := couleur des icônes fichier (`gray` par défaut);
- `AffIcoD` := booléen pour afficher les icônes dossier;
- `AffIcoF` := booléen pour afficher les icônes fichier;
- `EspaceV` := espace vertical entre les items (`0.15em` par défaut);
- `IcoD` := icône par défaut dossier;
- `IcoF` := icône par défaut fichier.

Les dossiers sont à déclarer sous la forme `<nom_dossier>,ADdos`.

Les fichiers sont à déclarer sous la forme `<nom_fichier>,ADfic`.

L'argument obligatoire, entre {...}, correspond à des options spécifiques, à passer en langage `forest`.

```
\begin{ArborDossiers}{ }
  [dossier,ADdos
    [sous-dossier,ADdos
      [sous-dossier,ADdos
        [fichier1,ADfic]
        [fichier2,ADfic]
      ]
    ]
  ]
  [fichier3,ADfic]
]
\end{ArborDossiers}
```

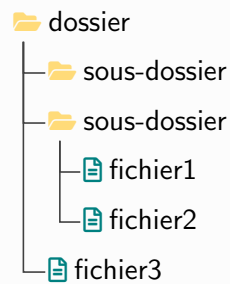


Le diagramme illustre l'arborescence générée par le code LaTeX ci-dessus. À la racine se trouve un dossier nommé 'dossier'. Sous ce dossier, il y a deux sous-dossiers. Le premier sous-dossier contient deux fichiers, 'fichier1' et 'fichier2'. Le second sous-dossier est vide. À côté du dossier principal, se trouve un fichier nommé 'fichier3'.

```

\begin{ArborDossiers}%
  [Police=\sffamily,CoulIcoD=yellow!50!pink,AffIcoF,CoulIcoF=teal,EspaceV=0.5em]%
  {}
  [dossier,ADdos
    [sous-dossier,ADdos
      [sous-dossier,ADdos
        [fichier1,ADfic]
        [fichier2,ADfic]
      ]
      [fichier3,ADfic]
    ]
  ]
\end{ArborDossiers}

```



Les styles sont définis de manière globale, et ils peuvent être redéfinis si besoin.

```

\tikzset{ADtrait/.style={thick,->,densely dashed,red,>=latex}}
\begin{ArborDossiers}%
  [Police=\sffamily,CoulIcoD=yellow!50!pink,AffIcoF,CoulIcoF=teal,EspaceV=0.5em]%
  {l sep=2em}
  [dossier,ADdos
    [sous-dossier,ADdos
      [sous-dossier,ADdos
        [fichier1,ADfic]
        [fichier2,ADfic]
      ]
      [fichier3,ADfic]
    ]
  ]
\end{ArborDossiers}

```

