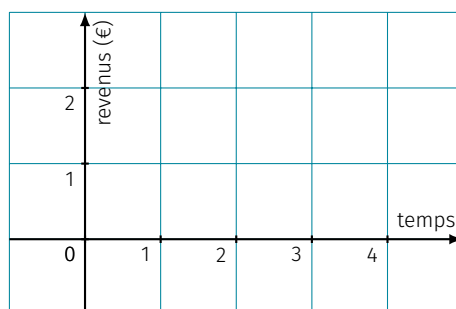


changelog

- 2023-09-09 : remplacement de la police Source Code Pro par la police JetBrains pour les passages en code.
- 2023-09-08 : ajout de `\reperenb{xmin}{ymin}{xmax}{ymax}{xlabel}{ylabel}`



- 2023-09-07 : changement majeur de l'appel de classe en 1^{re} ligne de document. rectifications et tests.

Options

Attention

Désormais les classes `nsibook`, `draftnsibook` et autres n'existent plus. Une seule classe est présente : la classe `nsi`.

Cependant, pour des raisons de rétrocompatibilité, ces anciennes classes sont encore incluses dans le répertoire `old` mais il est déconseillé de continuer à les utiliser car elles ne sont plus maintenues / mises à jour.

Il est possible de passer à la classe toutes les options *traditionnelles* des classes `book` ou `article` : `12pt` ou `11pt`, `a4paper` et *cætera*.

En plus de ces options traditionnelles

- `cours` sert à fabriquer des chapitres de cours;
- `exos` sert à fabriquer des feuilles d'exercices;
- `eval` sert à fabriquer des évaluations;
- `article` sert à fabriquer de cours documents (ressemble beaucoup à `exos`).

Exemple : création d'une feuille d'exercices

```
% Anciennement on écrivait
% \documentclass[12pt,a4paper]{nsiexos} % ou draftnsiexos
\documentclass[12pt,a4paper,exos]{nsi}
  \classe{\seconde 4}
  \titre{Feuille n°1}
  \begin{document}
    \maketitle
  \end{document}
```

Pour tous types de documents, `\maketitle` utilise `\titre` et `\classe`, sauf cours qui n'utilise pas cette dernière.

Compilation

Lorsque le document est compilé avec pdf \LaTeX , la compilation est plus rapide mais les polices de caractères de base sont utilisées.

Avec Lua \LaTeX , la compilation est plus lente mais les polices Fira Sans et Source Code Pro sont chargées.

L'option pdf \LaTeX est une bonne manière de préparer le travail, surtout sous Windows car la compilation est très lente.

Environnements

Définition

```
\begin{definition}[ : précision]
  contenu
\end{definition}
```

Définition : précision

contenu

Exemple

```
\begin{exemple}[ : précision]
  contenu
\end{exemple}
```

Exemple : précision

contenu

Propriété

```
\begin{propriete}[ : précision]
  contenu
\end{propriete}
```

Propriété : précision

contenu

Notation

```
\begin{notation}[ : précision]
  contenu
\end{notation}
```

Notation : précision

contenu

Méthode

```
\begin{methode}[ : précision]
  contenu
\end{methode}
```

Méthode : précision

contenu

Remarque

```
\begin{remarque}[ : précision]
    contenu
\end{remarque}
```

Remarque : précision

contenu

À retenir

```
\begin{aretenir}[ : précision]
    contenu
\end{aretenir}
```

À retenir : précision

contenu

Pour le code

```
\begin{pyc}
    \begin{minted}{python}
        def f(x: float) -> float:
            return 0.5 * x ** 2
    end{minted} % avec un \ devant
end{pyc}% avec un \ devant
```

Python

```
def f(x: float) -> float:
    return 0.5 * x ** 2
```

Je veux vous parler de la fonction `\mintinline{python}{print}` de
↪ `\textsc{Python}`.

Je veux vous parler de la fonction `print` de PYTHON.

Encadré coloré custom

```
\begin{encadrecolore}{Titre customisé de la couleur
↪ désirée}{UGLiDarkBlue}
    contenu
\end{encadrecolore}
```

Titre customisé de la couleur désirée

contenu

Environnements énumératifs

Liste ordonnée

```
\begin{enumerate}
    \item truc ;
    \item machin ;
    \item bidule.
\end{enumerate}
```

1. truc;
2. machin;
3. bidule

Liste non ordonnée

```
\begin{itemize}
  \item truc ;
  \item machin ;
  \item bidule.
\end{itemize}
```

- truc;
- machin;
- bidule.

QCM

Une question à choix multiples

```
\begin{qcm}
  \item Réponse 1
  \item Réponse 2
  \item Réponse 3
\end{qcm}
```

Une question à choix multiples

- ☐ a. Réponse 1
- ☐ b. Réponse 2
- ☐ c. Réponse 3

Couleurs

```
\color{UGLiPurple} UGLiPurple \\
\color{UGLiRed} UGLiRed \\
\color{UGLiOrange} UGLiOrange \\
\color{UGLiYellow} UGLiYellow \\
\color{UGLiGreen} UGLiGreen \\
\color{UGLiDarkGreen} UGLiDarkGreen \\
\color{UGLiBlue} UGLiBlue \\
\color{UGLiDarkBlue} UGLiDarkBlue
```

UGLiPurple
UGLiRed
UGLiOrange
UGLiYellow
UGLiGreen
UGLiDarkGreen
UGLiBlue
UGLiDarkBlue

Tables

Le style de table par défaut est sélectionnable avec `\tabdefault` (ou `\tabulardefault` pour rétrocompatibilité).

```
\tabdefault
\begin{tabular}{|c|c|c|}
  \hline
  Colonne 1 & Colonne 2 & Colonne 3 \\ \hline
  Valeur 1 & Valeur 2 & Valeur 3 \\ \hline
  Valeur 4 & Valeur 5 & Valeur 6 \\ \hline
  Valeur 7 & Valeur 8 & Valeur 9 \\ \hline
  Valeur 10 & Valeur 11 & Valeur 12 \\ \hline
  Valeur 13 & Valeur 14 & Valeur 15 \\ \hline
\end{tabular}
```

Colonne 1	Colonne 2	Colonne 3
Valeur 1	Valeur 2	Valeur 3
Valeur 4	Valeur 5	Valeur 6
Valeur 7	Valeur 8	Valeur 9
Valeur 10	Valeur 11	Valeur 12
Valeur 13	Valeur 14	Valeur 15

On peut styler les tables avec `\tabstyle[couleur]` (ou `\tabularstyle[couleur]` pour rétrocompatibilité)

```
\tabstyle[UGLiGreen]
\begin{tabular}{|c|c|c|}
  \hline
  Colonne 1 & Colonne 2 & Colonne 3 \\ \hline
  Valeur 1 & Valeur 2 & Valeur 3 \\ \hline
  Valeur 4 & Valeur 5 & Valeur 6 \\ \hline
  Valeur 7 & Valeur 8 & Valeur 9 \\ \hline
\end{tabular}
```

```
Valeur 10 & Valeur 11 & Valeur 12 \\hline
Valeur 13 & Valeur 14 & Valeur 15 \\hline
\end{tabular}
```

Colonne 1	Colonne 2	Colonne 3
Valeur 1	Valeur 2	Valeur 3
Valeur 4	Valeur 5	Valeur 6
Valeur 7	Valeur 8	Valeur 9
Valeur 10	Valeur 11	Valeur 12
Valeur 13	Valeur 14	Valeur 15

À l'intérieur d'un tableau stylé on peut utiliser la commande `\ccell` (ou `\ths` pour rétro-compatibilité) pour obtenir une cellule d'en-tête. Pour une cellule blanche, utiliser `\bcell`

```
\tabstyle[UGLiOrange]
\begin{tabular}{|c|c|c|}
\hline
\bcell & \ccell Colonne 2 & \ccell Colonne 3 \\hline
\ccell Valeur 1 & Valeur 2 & Valeur 3 \\hline
\ccell Valeur 4 & Valeur 5 & Valeur 6 \\hline
\ccell Valeur 7 & Valeur 8 & Valeur 9 \\hline
\end{tabular}
```

	Colonne 2	Colonne 3
Valeur 1	Valeur 2	Valeur 3
Valeur 4	Valeur 5	Valeur 6
Valeur 7	Valeur 8	Valeur 9

Lors de l'insertion d'une table dans un environnement, si le style des tables n'est pas `\tabdefault`, les couleurs de la table suivent celles de l'environnement :

```
\begin{exemple}[]
\begin{tabular}{|c|c|c|}
\hline
\bcell & \ccell Colonne 2 & \ccell Colonne 3 \\hline
\ccell Valeur 1 & Valeur 2 & Valeur 3 \\hline
\ccell Valeur 4 & Valeur 5 & Valeur 6 \\hline
\ccell Valeur 7 & Valeur 8 & Valeur 9 \\hline
\end{tabular}
\end{exemple}
```


Exemple

	Colonne 2	Colonne 3
Valeur 1	Valeur 2	Valeur 3
Valeur 4	Valeur 5	Valeur 6
Valeur 7	Valeur 8	Valeur 9

Mise en page

Avec les commandes

- `\picleft{fraction}{fichier}{texte}` (ou `\floatpictureleft` pour rétrocompatibilité);
- `\picleftc{fraction}{fichier}{legende}{texte}` (ou `\floatpictureleftcaption` pour rétrocompatibilité);
- Même chose à droite.

Du texte avant.\\

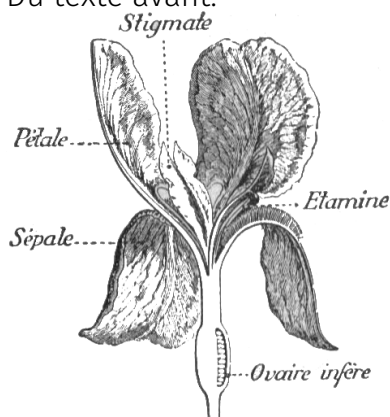
```
\floatpictureleft{0.3}{iris.png}{
```

Une image à gauche avec du texte à droite. En général on s'arrange
→ pour que le premier paramètre, qui est la fraction de la largeur
→ de la ligne occupée par l'image et la quantité de texte à droite
→ soient en harmonie sinon voici ce que cela donne.

```
}\par\medskip
```

Du texte après.

Du texte avant.



Une image à gauche avec du texte à droite. En général on s'arrange pour que le premier paramètre, qui est la fraction de la largeur de la ligne occupée par l'image et la quantité de texte à droite soient en harmonie sinon voici ce que cela donne.

Du texte après.

Du texte avant, qui peut prendre toute la ligne ou pas.\\

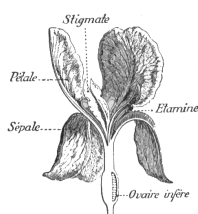
```
\floatpictureleft{0.15}{iris.png}{
```

Une image à gauche avec du texte à droite. En général on s'arrange
 ↪ pour que le premier paramètre, qui est la fraction de la largeur
 ↪ de la ligne occupée par l'image et la quantité de texte à droite
 ↪ soient en harmonie sinon on a vu ce que ça donne. Ce n'est pas
 ↪ catastrophique et cela peut même être désiré, mais en s'y prenant
 ↪ bien, voici à quoi on arrive. Ce n'est pas parfait mais je n'ai
 ↪ pas trouvé mieux !

```
}\par\medskip
```

Du texte après, qui peut prendre toute la ligne ou pas.

Du texte avant, qui peut prendre toute la ligne ou pas.



Une image à gauche avec du texte à droite. En général on s'arrange pour
 que le premier paramètre, qui est la fraction de la largeur de la ligne
 occupée par l'image et la quantité de texte à droite soient en harmonie
 sinon on a vu ce que ça donne. Ce n'est pas catastrophique et cela peut
 même être désiré, mais en s'y prenant bien, voici à quoi on arrive. Ce n'est
 pas parfait mais je n'ai pas trouvé mieux!

Du texte après, qui peut prendre toute la ligne ou pas.

Avec les commandes `\dleft{largeur gauche}{contenu gauche}{contenu droite}` et
`\dright{largeur droite}{contenu gauche}{contenu droite}`

```
\dleft{7cm}{
  \begin{tikzpicture}
    \draw[fill=UGLiGreen!10](0,0) rectangle(7,2);
  \end{tikzpicture}
}
{Vraiment sympa ce petit rectangle vert pastel à gauche de ce texte.}
```



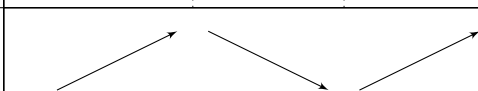
Vraiment sympa ce petit rectangle vert pastel à
 gauche de ce texte.

Tableaux de variations

```

\begin{center}
\begin{tikzpicture}
\tkzTabInit[color,lgt=2,espcl=2]
{\$x\$ /.7 ,\$f'(x)\$ /.7,\$f\$ /1.4}
{\$-\infty$, -1 ,5, \$+\infty\$ }
\tkzTabLine{,+ , z, -,z,+,}
\tkzTabVar{-/ ,+/, -/,+ /}
\end{tikzpicture}
\end{center}

```

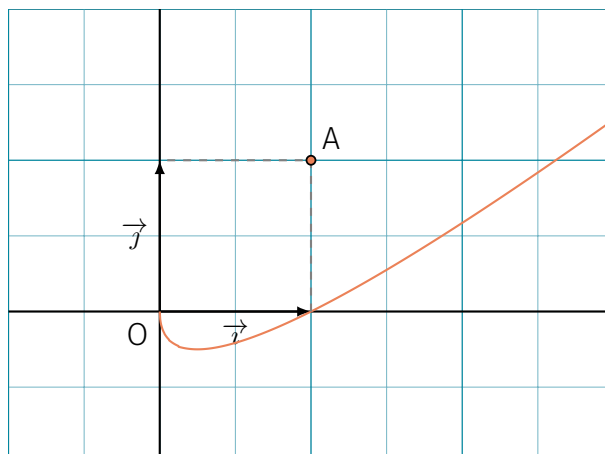
x	$-\infty$	-1	5	$+\infty$		
$f'(x)$		+	0	-	0	+
f						

Courbes représentatives

```

\begin{center}
\def\xmin{-1} \def\ymin{-1}\def\xmax{3}\def\ymax{2}
\def\F{\x-(\x)^{(.5)}}
\begin{tikzpicture}[scale=2]
\clip (\xmin,\ymin) rectangle (\xmax,\ymax);
\draw[fill = white] (\xmin,\ymin) rectangle (\xmax,\ymax);
\reperev{\xmin}{\ymin}{\xmax}{\ymax}
\draw[thick,domain=0:\xmax,samples=1000,UGLi0range,variable=\x]
↪ plot ({\x},{\F});
\point{1}{1}{A}
\end{tikzpicture}
\end{center}

```



Arbre de probabilités

```

\def\abun{A}
\def\alun{0,1}

\def\abdeux{\$\barmaj{A}\$}
\def\aldeux{\ldots}

\def\abtrois{\$A\cap B\$}
\def\altrois{\$p_A(B)\$}

\def\abquatre{\$A\cap\barmaj{B}\$}
\def\alquatre{0,7}

\def\abcinq{\$\barmaj{A}\cap B\$}
\def\alcinq{0,4}

\def\absix{\$\barmaj{A}\cap\barmaj{B}\$}
\def\alsix{\ldots}

\begin{center}
  \arbreproba
\end{center}

```

