I. Cas d'utilisation en LaTeX

1. Codes « *en ligne* »

La macro `\docilatex` permet de taper du code en ligne via un usage similaire à `\verb` comme dans les deux exemples suivants.

1. `\docilatex|$a^b = c$|` produit $a^b = c$.
2. `\docilatex+\docilatex|$a^b = c$|+` produit \docilatex|$a^b = c$| .

2. Codes tapés directement

Exemple 1 – Face à face

```
\begin{doclatex}
    $A = B + C$
\end{doclatex}
```

Ceci donne :

`$A = B + C$`	$A = B + C$

Exemple 2 – À la suite

Via `\begin{doclatex-flat} ... \end{doclatex-flat}`, on obtient un code à plat[1] comme ci-dessous.

```
$A = B + C$
```

$A = B + C$

Exemple 3 – Juste le code

`\begin{doclatex-alone} ... \end{doclatex-alone}` affiche le code seul[2] comme ci-après.

```
$A = B + C$
```

3. Codes importés

Pour les codes suivants, on considère un fichier `examples/listing-xyz.tex` dont le chemin est donné relativement au document présent. Le contenu de ce fichier est l'unique ligne `$x y z = 1$` .

Notez que les 1^res^ macros sont nommées de façon similaire aux environnements précédents en ajoutant le préfixe `input` tout en ignorant les tirets.

1. Le suffixe `flat` signifie « *plat* » en anglais.
2. Le suffixe `alone` signifie « *seul* » en anglais.

Remarque. *Il faut savoir que les macros imprimant automatiquement du texte tiennent code de la langue choisie lors du chargement du package* `bdoc`*.*

Exemple 1 – Face à face

```
\inputdoclatex{examples/listing-xyz.tex}
```

Ceci produit la mise en forme suivante.

```
$x y z = 1$
```

$xyz = 1$

Exemple 2 – À la suite

`\inputdoclatexflat` produit un code à plat comme ci-dessous.

```
$x y z = 1$
```

$xyz = 1$

Exemple 3 – Juste le code

`\inputdoclatexalone` sert à n'avoir que le code comme ci-après.

```
$x y z = 1$
```

Exemple 4 – Code suivi du rendu centré

Le rendu suivant est obtenu en utilisant `\inputdoclatexbefore`.

Début du rendu réel

```
$x y z = 1$
```

Rendu du code précédent.

$$xyz = 1$$

Fin du rendu réel

Il est possible de changer le texte entre le code et son rendu via un argument optionnel. Ainsi `\inputdoclatexbefore[Voici ce que cela donne.]{...}` aboutit au résultat suivant.

Début du rendu réel « personnalisé »

```
$x y z = 1$
```

Voici ce que cela donne.

$$xyz = 1$$

Fin du rendu réel « personnalisé »

Exemple 5 – Rendu centré suivi du code

Le rendu suivant [3], similaire au précédent, est obtenu en appellant `\inputdoclatexafter` au lieu de `\inputdoclatexbefore`.

Début du rendu réel

$$xyz = 1$$

Le rendu précédent a été obtenu via le code suivant.

```
$x y z = 1$
```

Fin du rendu réel

Via `\inputdoclatexafter[Cette formule se tape comme suit.]{...}` , on obtient le résultat ci-après.

Début du rendu réel « personnalisé »

$$xyz = 1$$

Cette formule se tape comme suit.

```
$x y z = 1$
```

Fin du rendu réel « personnalisé »

Exemple 6 – Code importé et son rendu réel

Pour un code et son rendu réel non centré, on utilisera `\inputdoclatexreal` qui va produire ce qui suit

3. Il faut savoir que le 1er espace vertical disgracieux vient de l'emploi de `\begin{center} ... \end{center}` en coulisse.

Début du rendu dans cette doc.

```
$x y z = 1$
```

Ceci donne :

Début du rendu réel

$xyz = 1$

Fin du rendu réel

Fin du rendu dans cette doc.

Là aussi le texte par défaut peut être « *personnalisé* ». Par exemple, ce qui suit a étét obtenu en tapant `\inputdoclatexreal[On obtient le rendu réel ci-après.]{...}` .

Début du rendu « **personnalisé** » *dans cette doc.*

```
$x y z = 1$
```

On obtient le rendu réel ci-après.

Début du rendu réel

$xyz = 1$

Fin du rendu réel

Fin du rendu « **personnalisé** » *dans cette doc.*