

Modèle pour les sujets d'exercices

08-08-2024

0.1.0

Typst 0.11.1

Mathieu AUCEJO

Typst est un modèle **Typst** pour la rédaction de sujets d'exercices ou d'exams

Table des matières

1. Qu'est-ce que Typst ?	1
2. Usage	2
2.1. Utilisation du template	2
2.2. Initilisation du modèle	2
2.3. Fonctions additionnelles	3
2.3.1. Environnements	3
2.3.2. Boîtes englobantes	4
2.3.3. Sous-figures	5
3. Feuille de route	5

1. Qu'est-ce que Typst ?

Typst est un nouveau langage de balise open source écrit en Rust et développé à partir de 2019 par deux étudiants allemands, Laurenz Mäde et Martin Haug. La version 0.1.0 a été publiée sur GitHub le 04 avril 2023¹.

Typst se présente comme un successeur de $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ plus moderne, rapide et simple d'utilisation. Parmi ses avantages, on peut citer :

- la compilation incrémentale ;
- des messages d'erreur clairs et compréhensibles ;
- un langage de programmation Turing-complet ;
- un système de style cohérent permettant de configurer aisément tous les aspects de son document (police, pagination, marges, ...) ;
- une communauté active et sympathique (serveur Discord pour le support, annonce de nouveaux paquets) ;
- un système de paquets simple d'utilisation (pour rechercher ou voir la liste des paquets, n'hésitez pas à visiter [Typst: Universe](#)) ;
- des extensions pour VS Code existent, comme `Typst LSP` ou `Typst preview`, pour avoir des fonctionnalités similaires à `LaTeX Workshop`.

Pour finir, la documentation de **Typst** est suffisamment bien écrite et détaillée pour permettre de créer rapidement ses propres documents. Il faut compter moins d'une heure pour prendre en

¹Adresse du dépôt GitHub : <https://github.com/typst/typst>

1. Qu'est-ce que Typst ?

main la syntaxe (sans mentir 😊). Pour accéder à la documentation, suivez ce [lien](#). Pour faciliter la transition de L^AT_EX vers Typst, un guide est disponible [ici](#).

2. Usage

2.1. Utilisation du template

Pour utiliser le modèle, il faut l'importer dans votre fichier principal typ en utilisant la commande suivante.

```
#import "./tutorial.typ": *
```

Si vous décomposez votre document en différents fichiers, il faut insérer la commande précédente en préambule de chaque fichier.

2.2. Initilisation du modèle

Après avoir importé le modèle, celui doit être initialisé en appliquant une règle d'affichage (show rule) avec la commande `#tutorial()` en passant les options nécessaires avec l'instruction `with` dans votre fichier principal typ :

```
#show tutorial.with(  
  ...  
)
```

Le modèle `#tutorial()` possède un certain nombre de paramètres permettant de personnaliser le document. Voici la liste des paramètres disponibles :

`#tutorial(<title>: none, <subtitle>: none, <docfont>: "Lato", <docfontmath>: "Lete Sans Math") [<body>]`

Argument

`<title>: none`

str

Titre du document

Argument

`<subtitle>: ()`

str

Sous-titre du document

Argument

`<docfont>: none`

str

Police de caractères pour le corps du texte

Argument

`<docfontmath>: none`

str

Police de caractères pour les équations mathématiques

2. Usage

2.3. Fonctions additionnelles

Le modèle [Tutorial](#) fournit un certain nombre de fonctions additionnelles pour faciliter la rédaction de votre document.

2.3.1. Environnements

Exercice

Pour créer un nouvel exercice, il suffit d'utiliser la commande `#exercice()` :

```
#exercice("Titre de l'exercice")[Texte de l'exercice]
```

La numérotation des exercices est automatique.

Question

Pour ajouter une question à un exercice, il suffit d'utiliser la commande `#question()` :

```
#question[Texte de la question]
```

La numérotation des questions est automatique et est réinitialisée pour chaque exercice.

Correction

Pour ajouter une correction à une question, il suffit d'utiliser la commande `#correction()` :

Exemple

```
// À mettre au début du document
// Mettre le paramètre `false` pour masquer les corrections
#let corr = true

// À mettre à la suite de la question
#let rep = [Correction]
#correction(corr, rep)
```

Réponse

Correction

2. Usage

2.3.2. Boîtes englobantes

Les boîtes englobantes sont des éléments graphiques permettant de mettre en avant des informations importantes. Le modèle [Tutorial](#) en propose actuelle trois types : obj, reco et info.

1. La boîte obj est utilisée pour mettre en avant les objectifs des exercices ou de l'examen.

Exemple

```
#obj[
+ Objectif 1
+ Objectif 2
]
```

Objectifs pédagogiques

1. Objectif 1
2. Objectif 2

2. La boîte reco est utilisée pour mettre en avant les recommandations pour la réalisation des exercices ou de l'examen.

Exemple

```
#reco[
#lorem(20)
]
```


Recommandations

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua quaerat.

3. La boîte info est utilisée pour mettre en avant des informations importantes.

Exemple

```
#info[
#lorem(20)
]
```

 Remarque

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua quaerat.

2. Usage

2.3.3. Sous-figures

Typst ne dispose pas actuellement de mécanismes permettant de gérer les sous-figures (numérotation et référencement). Pour pallier ce manque, le modèle intègre une fonction `#subfigure()` permettant de gérer les sous-figures de manière adaptée. Cette fonction encapsule la fonction `#subpar.grid()` du package `subpar`.

```
#subfigure(  
  figure(image("image1.png"), caption: []), <figa>,  
  figure(image("image2.png"), caption: []), <figb>,  
  columns: (1fr, 1fr),  
  caption: [(a) Première image and (b) Seconde image],  
  label: <fig>  
)
```

3. Feuille de route

Le modèle **Tutorial** est en cours de développement. Certaines fonctionnalités devront être implémentées dans les prochaines versions. Voici la liste des fonctionnalités actuelles et à venir :

- ☒ Mise en place du modèle de base
- ☐ Ajout de boîtes englobantes (warning, tip, important)
- ☐ Ajout de zones de réponses