T-DOC

Un écosystème pour la génération de pages Web pour enseignants de mathématiques et informatique

Travail personnel - Informatique+30

Caroline Blank

Avril 2024

Supervisé par :

Prof. Dr. Jacques Pasquier
Software Engineering Group

UNI FR
UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

Groupe Génie Logiciel
Département d'Informatique
Université de Fribourg (Suisse)

Software Engineering Group

Résumé

Le but du projet est de faciliter la réutilisation de documents existants pour les mettre à disposition des élèves sur le Web. L'enseignant enrichit ses documents en LaTeX par du code Python, des vidéos et des fenêtres Geogebra.

Le coeur du projet est constitué d'un moteur de rendu qui transforme le document rédigé en LaTeX, complété par des balises personnalisées, en une page HTML. Pour le cours de mathématiques, l'élève pourra visionner dans un navigateur une page qui contient des parties de texte et de contenu mathématique, par exemple, de la théorie, des exercices ou des solutions, une illustration géométrique en utilisant une fenêtre Geogebra, ainsi que des compléments de théorie ou des exemples détaillés en format vidéo. Dans le cadre du cours d'informatique, les pages pourront, en plus, afficher du code surligné afin de faciliter la compréhension et exécuter du code directement dans le navigateur. Cela donne une séquence d'enseignement en autonomie pour la découverte d'une nouvelle notion, la possibilité de revoir une notion pas bien comprise ou tout simplement de réviser pour un examen.

Table des matières

Introduction

T-doc (teaching docs) constitue une mise en pratique des connaissances acquises lors de la formation universitaire en informatique. Le projet a été choisi de manière répondre à un besoin constaté depuis quelques années et pourra être utilisé dans mon enseignement.

En tant qu'enseignante au collège, la préparation des supports de cours prend un temps considérable, notamment la rédaction de document LaTeX. Depuis la rentrée 2022-2023, les élèves des collèges fribourgeois sont passés en mode BYOD[1], ils ont donc un ordinateur à portée de main à chaque cours. Actuellement, beaucoup d'enseignants se limitent à transmettre leurs cours en format PDF par manque de temps et de motivation à faire mieux. Mais un polycopié en format PDF n'apporte rien de plus aux élèves que sa version papier. Le Web regorge de matériel pédagogique que ce soient des vidéos, des exercices en ligne, des outils pour faire de la géométrie interactive, etc. Malheureusement il est difficile pour les élèves de trouver les bonnes ressources adaptées à leur niveau. C'est ainsi qu'est apparue l'idée de créer un outil qui combine du LaTeX, du Python, des images, des vidéos et des figures Geogebra pour un rendu en une page HTML qui peut être visualisée et avec laquelle on peut interagir dans un navigateur. Les élèves profitent ainsi, en plus du polycopié, d'outils à disposition pour la découverte d'une nouvelle notion, la possibilité de revoir une notion pas bien comprise ou tout simplement de réviser pour un examen.

Le coeur du projet est constitué d'un moteur de rendu qui transforme le document d'origine contenant du LaTeX en une page HTML. Pour le cours de mathématiques, l'élève pourra visionner dans un navigateur une page qui contient des parties de texte et de contenu mathématique, par exemple, de la théorie, des exercices ou des solutions, une illustration géométrique en utilisant une fenêtre Geogebra, ainsi que des compléments de théorie ou des exemples détaillés en format vidéo. Dans le cadre du cours d'informatique, les pages pourront, en plus, afficher du code surligné afin de faciliter la compréhension et exécuter ce code directement dans le navigateur. Cela donne des séquences d'enseignement en autonomie.

1. Bring Your Own Device

1 Cahier des charges

Cette section définit un cahier des charges des fonctionnalités. Elle précise le profil des utilisateurs cibles. Ensuite, en se mettant à leur place, il faut déterminer les fonctionnalités nécessaires et celles qui sont plutôt optionnelles et qui seront développées plus tard.

1.1 Personas

Pour ce projet, on considère deux personas : les enseignants et les élèves.

Cet outil s'adresse à des enseignants de mathématiques du collège qui ont donc des connaissances en LaTeX et des enseignants d'informatique qui, s'ils ne connaissent pas LaTeX, peuvent facilement acquérir les bases nécessaires.

L'enseignant doit pouvoir :

- adapter rapidement ses documents LaTeX pour la création de pages Web,
- créer de nouveaux documents,
- éditer des documents existants,
- visualiser le rendu des documents de manière itérative.

L'enseignant utilisera un éditeur de son choix pour créer les documents.

L'élève doit pouvoir :

- apprendre une nouvelle notion,
- revoir une notion pas bien comprise en classe,
- réviser pour un examen.

L'élève aura à disposition des séquences d'enseignement en autonomie sous forme de pages Web. Il aura donc besoin d'un navigateur et de l'URL pour accéder à la page.

1.2 LaTeX

LaTeX est un langage qui permet de structurer un document et son contenu. La rédaction des documents se fait sous forme de texte pur avec des balises pour définir la structure, ainsi n'importe quel éditeur peut être utilisé. Ensuite, le code source est traité par un compilateur, par exemple xelatex ou luatex, qui va générer, en général, un document PDF dont la mise en page est effectuée automatiquement. Il est beaucoup utilisé par les enseignants de mathématiques, car il permet un très bon rendu des formules mathématiques. En effet, TeX, le précurseur de LaTeX, a été créé dans les années 70 par Donald Knuth dans ce but.

Les enseignants de mathématiques du collège rédigent leurs documents de cours en LaTeX, car il permet d'écrire "facilement" du contenu mathématique. Il est donc essentiel que ce projet intègre LaTeX. Il se focalise, dans un premier temps, sur la compatibilité des commandes standards utilisées pour l'enseignement des mathématiques, c'est-à-dire les formules (environnement mathématique), les symboles, les listes ordonnées ou non-ordonnées (enumerate et itemize), l'affichage avec des colonnes multiples (multicols), les tableaux (array et tabular) et les images. Le support pour des commandes moins utilisées sera ajouté par la suite.

1.3 Multimédia

Depuis plusieurs années, l'apprentissage au moyen de vidéos est très apprécié, notamment par les adolescents. Cela ne remplace pas un enseignement en classe par un professeur qui est disponible et pourra répondre aux questions, mais cela permet d'apprendre à son rythme, seul et à n'importe quel moment du jour ou de la nuit. Il est souvent plus simple pour les élèves de regarder une vidéo que de lire un polycopié. Pour ces raisons, l'ajout de vidéo était une nécessité. Comme il existe beaucoup de vidéos de qualité d'enseignants de mathématiques sur Youtube, notamment celles d'Yvan Monka[2], le projet n'intégrera, dans un premier temps, que des vidéos Youtube.

Toucher et tester facilitent les apprentissages. Pour les mathématiques, Geogebra est un logiciel interactif de géométrie, d'analyse, de statistique et de calcul différentiel. En l'intégrant dans ce projet, les élèves peuvent expérimenter "en live" différentes approches.

2. https://www.youtube.com/@YMONKA

1.4 Programmation

Le langage de programmation utilisé dans les collèges fribourgeois est principalement le Python. Lors de l'enseignement de la programmation, il y a plusieurs contraintes à suivre :

- Les élèves débutent en programmation, il leur est difficile de lire et comprendre du code. Un surlignement de code correct, si possible proche des éditeurs standards, facilite la compréhension par les élèves.
- L'exécution de code, directement sur la page, permet aux élèves d'essayer de comprendre un programme et ensuite de l'exécuter pour valider ou non leur réponse. Cela leur permet d'avoir un feedback instantané sans que l'enseignant doive valider la réponse.
- Le module Turtle permet de faire de la programmation visuelle. Le dessin lui-même permet de valider ou non la réussite de l'exercice. Ce module est très souvent utilisé par les enseignants du collège, il est donc nécessaire de pouvoir aussi exécuter du code qui utilise ce module.

2 Design et réalisation

Dans cette partie, les différents outils utilisés et les choix effectués seront présentés et expliqués. Le code source de ce projet est disponible sur GitHub : <https://github.com/t-docs/t-doc>.

2.1 Framework Web

Comme j'enseigne le Python à mes élèves, je souhaitais l'utiliser pour ce projet et peut-être l'utiliser avec les élèves dans le cadre de l'option complémentaire d'informatique, afin qu'ils développent eux-mêmes une application Web. Parmi les nombreux outils pour développer une application Web, mon choix s'est porté sur Django[3], un framework Web en Python, parce que je l'ai déjà utilisé dans un autre projet et qu'il a de nombreux avantages, entre-autres :

- Il est gratuit et open source.
- Il intègre un serveur Web qui permet de développer et tester une application en temps réel sans déploiement.
- Il fournit une fonctionnalité de cache flexible.

3. https://www.djangoproject.com/

2.2 Côté client versus côté serveur

Lors du développement d'une application Web, la question du rendu côté client ou côté serveur se pose. Chacun ayant ses avantages et inconvénients. L'avantage majeur du rendu côté client est la sécurité. En effet, comme l'exécution de code se fait directement dans le navigateur et reste dans le navigateur, les risques sont moindres, d'autant plus que les navigateurs sont développés pour assurer un maximum de sécurité. L'avantage du rendu côté serveur est la facilité, car le rendu se fait en une fois. Mais cela engendre plus de travail pour le déploiement, car il faut installer et maintenir un serveur. Cette question se posera fréquemment lorsqu'il faudra prendre des décisions, car il est généralement possible de développer l'une ou l'autre option, avec des niveaux de complexité variables en fonction des outils sélectionnés.

2.3 Format du document d'origine

L'application doit générer du HTML à partir d'un document pouvant contenir du LaTeX, par conséquent, choisir HTML ou LaTeX comme format du document source semble une évidence. Dans un premier temps, HTML a été privilégié, car il s'agit d'un format plus cohérent et activement développé, mais cette option a été abandonnée par la suite[4].

L'utilisation de LaTeX comme format d'origine oblige à faire le rendu côté serveur, puisque le navigateur ne comprend pas le LaTeX. Même si cela implique de devoir gérer un serveur, ce choix à l'avantage de faciliter le travail de l'utilisateur. En effet, les enseignants pourront utiliser leurs documents rédigés en LaTeX moyennant quelques modifications, afin d'ajouter les fonctionnalités interactives.

4. voir Alternatives considérées

2.4 Génération d'une page HTML à partir d'un document LaTeX

Il existe différents outils [5] qui permettent de convertir du LaTeX en HTML. La qualité du rendu, notamment des formules mathématiques, est très variable. Au niveau de l'utilisation, certains permettent de générer des fragments de LaTeX, mais la plupart requièrent la génération d'un document complet.

Lors de mes recherches, deux outils ont particulièrement retenu mon attention :

1. LaTeX.js[6], une librairie JavaScript.
2. Make4ht[7], un package contenu dans la plupart des distributions LaTeX.

Le premier outil a été testé, mais finalement abandonné[8], car il ne permettait pas le rendu de toutes les fonctionnalités LaTeX souhaitées.

Make4ht est un système de construction simplifié pour TeX4ht[9] lui-même, un convertisseur de TeX en HTML. Make4ht contient un outil en ligne de commande qui gère le processus de conversion configurable. Contrairement à d'autres convertisseurs comme LaTeX2HTML, qui convertissent le document TeX en HTML directement, TeX4ht utilise le fichier DVI[10] résultant de la compilation "normale" en entrée. Cela permet une reproduction plus fidèle du document.

Pour générer le rendu depuis Django, la commande suivante est exécutée :

```
1 $ make4ht -c latex.cfg -j "output" "monfichier.tex" "mathjax"
```

Code 1: Ligne de commande pour le rendu

Cette commande génère deux documents : le fichier HTML (output.html) et le fichier CSS (output.css) qui définit la mise en page. Afin de simplifier le transfert au navigateur, le CSS est ensuite intégré directement dans l'entête du document HTML.

5. voir Annexe 2 : Outils de rendu LaTeX en HTML
6. https://latex.js.org/
7. https://ctan.org/pkg/make4ht?lang=en
8. voir Alternatives considérées
9. voir https://tug.org/tex4ht/
10. DeVice-Independent

2.5 Affichage de contenu mathématique

Make4ht permet de configurer le rendu de contenu mathématique. Par défaut, le rendu se fait en MathML, mais la qualité n'est pas acceptable. Il est aussi possible de générer des images SVG. Même si la qualité du rendu est très bonne, ces images ne sont pas stockées directement dans le HTML, mais dans des fichiers séparés. Cela complique le transfert au navigateur, car il faudrait soit garder toutes les références et transmettre plusieurs fichiers, soit inclure les images directement dans le fichier HTML. De plus, les formules sous forme d'image empêchent leur recherche. La solution retenue qui est la plus simple, est d'utiliser MathJax[11]. MathJax est un moteur d'affichage JavaScript open source pour les notations LaTeX, MathML et AsciiMath qui fonctionne dans tous les navigateurs modernes. Il ne nécessite aucune installation.

2.6 Ajout de balises personnalisées

LaTeX permet de créer ses propres macros afin de faciliter la rédaction de documents. Cette possibilité sera utilisée pour la définition de nouvelles balises personnalisées, ainsi l'enseignant pourra intégrer facilement les nouveaux éléments (vidéos, code, solutions, …) dans son document LaTeX.

Ces balises personnalisées seront traitées lors de la génération du document avec make4ht à l'aide du fichier de configuration. Celui-ci remplace la balise \begin{document} du document LaTeX par le contenu du fichier de configuration qui contient, par exemple, des nouvelles commandes ou des nouveaux environnements.

Le rendu des balises personnalisées utilise une combinaison de trois techniques :

- Les transformations se font directement lors du rendu LaTeX :
  Les balises sont remplacées directement par du code HTML. Ce procédé est utilisé pour les images, les vidéos et les fenêtres Geogebra.
- Les manipulations se font grâce à du CSS :
  Ce procédé est utilisé pour l'affichage des solutions. Lors du rendu LaTeX, les balises sont remplacées par des balises avec une classe CSS spécifique. Ainsi le CSS pourra traiter les blocs de solutions correctement.
- Les manipulations se font côté client en JavaScript :
  Lors du rendu LaTeX, les balises sont remplacées par des balises HTML avec une classe CSS spécifique. Cela permettra à JavaScript de les retrouver et de surligner le code ou de l'exécuter correctement.

11. https://docs.mathjax.org/en/latest/index.html

2.6.1 Vidéo

LaTeX contient des packages pour ajouter des vidéos à un PDF, mais cela ne permet pas d'afficher et de pouvoir visionner des vidéos Youtube sur un site Web à partir de sa référence. Cette transformation se fait directement lors du rendu LaTeX, grâce à une balise personnalisée \video{référence} qui est remplacée par le code HTML d'intégration de vidéos, en tenant compte de la référence passée en paramètre et de la taille de la vidéo. La taille est optionnelle et par défaut à 50% de la largeur du texte.

2.6.2 Geogebra

L'affichage d'une fenêtre Geogebra[12] se fait exactement de la même manière que pour les vidéos. Le document doit être, au préalable, sauvegardé en ligne. Par défaut, la taille de la fenêtre est de 100% de la largeur du texte.

2.6.3 Code source

Le surlignement de blocs facilite la lecture et la compréhension. En LaTeX, le package lstlisting permet de le faire facilement, car les principaux langages sont prédéfinis. Toutefois, lors de la génération du document HTML, make4ht ajoute des balises pour gérer l'indentation. Cela a pour conséquence que ce n'est plus un bloc de code et donc qu'il est impossible de le réutiliser, par exemple pour l'exécuter et afficher le résultat. La solution à ce problème est d'utiliser le package verbatim qui affiche exactement ce qui est écrit, c'est-à-dire qu'il garde l'indentation et les retours à la ligne, mais surtout il conserve le code dans une seule et même balise, ce qui permet de le réutiliser.

Le problème de l'indentation étant réglé, il faut maintenant surligner le code. Pour cela, il existe différents outils utilisant du JavaScript. Highlight.js[13] a été choisi pour sa simplicité d'utilisation. Il permet d'afficher 192 langages avec différents thèmes et fonctionne avec n'importe quel balisage HTML. Il a une fonctionnalité de détection automatique du langage, mais celle-ci ne fonctionne pas toujours correctement, particulièrement lorsque les blocs de code sont courts, comme c'est le cas, dans les exercices que je propose. Par conséquent, l'utilisateur devra toujours indiquer le langage utilisé dans le document LaTeX.

Pour utiliser highlight.js, il faut que le code se trouve dans un conteneur HTML .

12. https://www.geogebra.org/
13. https://highlightjs.org/

2.6.4 Exécution de code

Afin de s'assurer que l'élève a compris le code donné, il doit pouvoir l'exécuter dans le navigateur. Skulpt[14], un outil qui compile du Python en utilisant Javascript, offre une solution simple. En réalité, Skulpt traduit le code écrit en Python en code JavaScript, cela s'effectue côté client, donc directement dans le navigateur.

Il y a deux types d'exécution :

- le résultat est affiché sur la console,
- le code qui utilise le module Turtle[15] affiche un dessin dans un canevas graphique.

Skulpt permet de gérer les deux types d'exécution. L'enseignant détermine si le code pourra être exécuté grâce aux paramètres "interactive" ou "interactive_turtle", s'il utilise le module Turtle.

2.6.5 Solution

Pour permettre aux élèves de travailler en autonomie, il est nécessaire d'y inclure les solutions. L'idée est d'intégrer ces solutions au document LaTeX et l'élève pourra les afficher sur demande. Les solutions sont placées dans des balises HTML , ainsi elles pourront être manipulées par la suite par le navigateur.

L'affichage ou non des solutions se fait au moyen de CSS. Le principe pour l'affichage des solutions est le suivant : chaque solution a une checkbox invisible associée, si celle-ci est cochée, la solution sera affichée. Par défaut, les solutions ne sont pas affichées.

2.7 Sécurité

L'utilisation de LaTeX comme format d'origine et le choix de faire le rendu du côté serveur amène la question de la sécurité. Le navigateur transmet au serveur du code LaTeX et celui-ci peut contenir des macros, par exemple \input{mon_fichier} ou \lstinputlisting{code_source.py}, qui peuvent lire et écrire des fichiers arbitraires. Cela ouvre un accès au serveur et pose un problème de sécurité. En effet une personne pourrait rédiger un document LaTeX contenant une macro qui renvoie à un code malveillant et lors de la génération ce code pourrait s'exécuter. La résolution de ce problème est intéressante et représente un vrai défi. Cependant en raison des contraintes de temps pour ce projet, cette partie sera réalisée plus tard. En attendant l'enseignant assurera la gestion du serveur et la génération de ses propres documents, offrant ainsi une solution temporaire à ce problème.

14. https://skulpt.org/
15. https://docs.python.org/3/library/turtle.html

2.8 Cache

La génération de code HTML avec make4ht, même de documents simples, prend plus de 10 secondes. En effet, make4ht effectue le rendu trois fois, car certaines informations dépendent du rendu final, comme la table des matières d'un document, le nombre de pages total ou les références croisées. Malheureusement une latence de plusieurs secondes n'est pas acceptable pour les utilisateurs d'un site Web. Deux solutions sont implémentées pour y remédier :

- L'enseignant peut utiliser une version "draft" lors de la génération du code HTML, ainsi make4ht exécute le fichier une seule fois. Cela permet de réduire le temps de chargement à environ 4 secondes. Mais cette fonction, pratique lors de la création ou l'édition de documents, ne peut pas être utilisée dans la version finale.
- L'utilisation d'un cache persistant permet de diminuer de manière significative le temps de chargement de la page. Les pages créées pour ce projet étant statiques, il suffit de les générer à la première utilisation et ensuite de les sauvegarder sur le serveur. Si la page est modifiée, elle sera à nouveau générée.

Django intègre un système de cache sur système de fichiers qui stocke les fichiers dans le répertoire "cache", ainsi même si le serveur est arrêté, les fichiers seront préservés. Chaque document sera associé à une clé unique qui détermine sa version. La clé dépend des paramètres suivants :

1. Un "seed" utilisé pour l'invalidation du cache. Le "seed", défini dans le fichier settings.py du projet Django, permet l'effacement de tout le cache lors de changements qui concernent tous les documents, par exemple une mise à jour de LaTeX.
2. Le contenu du fichier LaTeX.
3. Le contenu du fichier de configuration.
4. Le mode : normal ou draft.

Lors du chargement de la page, le programme détermine la clé en fonction des différents éléments ci-dessus. Ensuite, il vérifie dans le cache si cette version de la page existe déjà. Si c'est le cas, il l'envoie directement au navigateur, sinon il génère la nouvelle page, la stocke et la transmet au navigateur.
Ainsi l'affichage d'une page pour un élève ne prend que quelques millisecondes.

3 Démonstration d'utilisation - Mode d'emploi

L'enseignant qui souhaite utiliser cette application, doit pouvoir créer des documents LaTeX avec une structure correcte. Pour cela, il a besoin :

- de connaissances de base de LaTeX,
- d'un éditeur de texte,
- de connaître les balises personnalisées qui peuvent être ajoutées,
- d'installer le serveur Web qui s'occupera de la génération et du stockage des documents.

3.1 Serveur Web

Pour créer le serveur Web dans un conteneur Docker[16], il faut :

1. Installer docker : https://docs.docker.com/desktop/
2. Cloner le projet : https://github.com/t-docs/t-doc.git
3. Se déplacer dans le répertoire t-doc/server
4. Lancer la commande :
   $ docker build -t t-doc-image .
   Cette commande crée une image pour le conteneur avec tous les packages nécessaires (Python3, Texlive, Django).
5. Créer une image à partir du Dockerfile :
   $ docker run -it –rm –mount type=bind,source='CheminAccèsProjet',target=/repo,readonly –publish 8000 :8000 t-doc-image
   Cette commande démarre un conteneur Docker à partir de l'image créée précédemment. Elle construit un pont entre le répertoire dans lequel le projet a été cloné et le conteneur, cela permet d'ajouter et d'éditer des documents localement, c'est-à-dire l'extérieur du conteneur.
   Le serveur Web démarre automatiquement.
6. Pour accéder localement aux pages d'exemples, utiliser l'adresse suivante :
   http ://localhost :8000/doc/Informatique/Exercices-Python
   http ://localhost :8000/doc/Algèbre/Fonctions
7. Déployer le serveur pour permettre l'accès depuis l'extérieur.

16. https://docs.docker.com/

3.2 Édition et création de documents

- Tous les documents se trouvent dans le répertoire t-doc/Documents. Ce répertoire peut être hiérarchisé. Cette hiérarchie sera utilisée pour l'affichage des documents dans le navigateur. Exemple : le document Trigonométrie.tex qui se trouve dans le répertoire Géométrie sera appelé avec l'URL suivante :
  http ://localhost :8000/doc/Géométrie/Trigonométrie
- Les images doivent être impérativement sauvegardées, dans le même répertoire, dans un dossier "images".
- Le fichier Template.tex, disponible dans le répertoire Documents, fournit un point de départ pour la création de nouveaux documents.
- L'utilisation des différentes balises personnalisées est expliquée dans le document Tutoriel.tex accessible à l'adresse suivante :
  http ://localhost :8000/doc/Tutoriel
- S'il y a des erreurs dans le document, les erreurs de make4ht sont affichées à la place de la page Web.

```
[ERROR]   htlatex: Compilation errors in the htlatex run
[ERROR]   htlatex: Filename     Line    Message
[ERROR]   htlatex: /tmp/lua_shTJ41      12      Missing $ inserted.
[WARNING] domfilter: DOM parsing of output.html failed:
[WARNING] domfilter: /usr/share/texmf-dist/tex/luatex/luaxml/luaxml-mod-xml.lua:175: Unbalanced Tag (/p) [char=11354]
```

Fig. 1 : Exemple de message d'erreur affiché dans le navigateur

3.3 Structure du document LaTeX

La structure du document LaTeX ne diffère pas beaucoup de la structure standard. Pour simplifier la rédaction et surtout alléger l'écriture de l'entête du document, la liste des packages usuels de LaTeX se trouve dans le fichier commonpackages.sty.

```
1  \ProvidesPackage{commonpackages}
2
3  \usepackage[french]{babel}
4  \usepackage[T1]{fontenc}
5  \usepackage{amssymb,amsmath,amsfonts, amsfonts}
6  \usepackage{enumitem}
7  \usepackage{multicol}
8  \usepackage{graphicx}
9  \usepackage{media9}
10 \usepackage{verbatim}
11 \usepackage{hyperref}
```

```
\usepackage{xcolor}

\endinput
```

Code 2: Fichier commonpackages.sty

Voici la structure de base du document avec l'import du package commonpackage et la définition et la création du titre de la page.

```
\documentclass[a4paper,11pt]{article}
\usepackage{commonpackages}

\begin{document}
\title{Noter le titre ici}
\date{}
\maketitle
...
\end{document}
```

Code 3: Structure de base du document LaTeX

Il est nécessaire de définir le titre et d'appeler la commande \maketitle, car celle-ci permet de définir le titre du document et le nom de l'onglet de la page HTML.

3.4 Balises personnalisées

La version Web de cette section se trouve à la page suivante : http://t-doc.org/doc/Tutoriel.

Afin de pouvoir ajouter des vidéos Youtube, une fenêtre Geogeabra, des images, du code et des solutions, des balises personnalisées peuvent être ajoutée au document LaTeX. Leur fonctionnement sera expliqué en détail au moyen d'exemples.

3.4.1 Vidéos Youtube

Pour ajouter une vidéo Youtube, il suffit de connaître sa référence et d'utiliser la balise suivante : \video{référence}

```
\documentclass[a4paper,11pt]{article}
\usepackage{commonpackages}

\begin{document}
\title{Produits remarquables}
\date{}
\maketitle
Cette vidéo explique comment développer des produits de polynômes en
   utilisant les produits remarquables.\par
```

```
\video{U98Tk89SJ5M}
\end{document}
```

Code 4: Code LaTeX avec ajout d'une vidéo Youtube

Fig. 2 : Rendu de la page HTML avec une vidéo

Par défaut, la taille de la fenêtre de la vidéo est 50% de la largeur de la page. Il est possible de changer la dimension en ajoutant un paramètre optionnel, celui-ci doit être noté entre crochets et avant la référence de la vidéo :

```
\video[100]{U98Tk89SJ5M}
```

Code 5: Code LaTeX pour une vidéo avec une dimension de 100%

Les vidéos peuvent être visionnées en plein écran.

3.4.2 Fenêtres Geogebra

Pour ajouter une fenêtre Geogebra, il faut tout d'abord créer le document Geogebra et le sauvegarder en ligne[17]. Ensuite, il pourra être intégré au document LaTeX au moyen de la balise suivante : \geogebra{référence}. La structure est la même que pour les vidéos Youtube.

```
\documentclass[a4paper,11pt]{article}
\usepackage{commonpackages}

\begin{document}
\title{Équation d'une droite}
\date{}
\maketitle
```

17. https://www.geogebra.org/?lang=fr

```
La fenêtre Geogebra permet à l'élève d'observer comment change la droite en
   fonction de a qui représente la pente et b qui est l'ordonnée à
   l'origine:\par
\geogebra{esdhdhzd}
\end{document}
```

Code 6: Code LaTeX avec ajout d'une fenêtre Geogebra

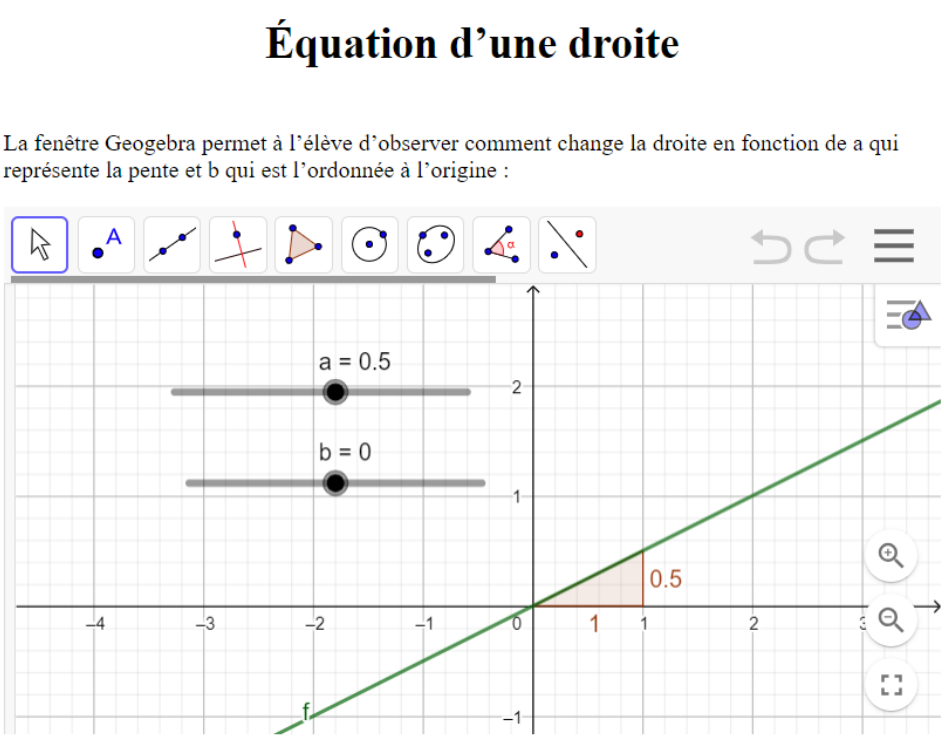


FIG. 3 : Rendu de la page HTML avec une fenêtre Geogebra

Par défaut, la taille de la fenêtre Geogebra est 100% de la largeur de la page, afin d'optimiser l'utilisation pour l'élève. Il est possible de changer la dimension en ajoutant un paramètre optionnel, celui-ci doit être noté entre crochets et avant la référence du document Geogebra :

```
\geogebra[75]{esdhdhzd}
```

Code 7: Code LaTeX pour une fenêtre Geogebra avec une dimension de 75%

Geogebra peut être utilisé en plein écran.

3.4.3 Images

L'ajout d'images ne nécessite pas la création d'une balise personnalisée, toutefois pour que l'affichage soit correct au moment du rendu, il est nécessaire de définir les dimensions de celle-ci en fonction de la largeur du texte (\textwidth) :

```
\includegraphics[width=0.8\textwidth]{images/trianglerect.png}
```

Code 8: Code LaTeX avec une image dont la largeur sera 80% de la largeur du texte

3.4.4 Code source

Pour afficher du code source, celui-ci sera simplement inclus dans un environnement code, c'est-à-dire qu'il sera délimité par les balises \begin{code} et \end{code}. Afin que le surlignement se fasse correctement, il est obligatoire de définir le langage comme paramètre.

```
\documentclass[a4paper,11pt]{article}
\usepackage{commonpackages}

\begin{document}
\title{Exemple de code en Python}
\date{}
\maketitle
\begin{code}{python}
for i in range(4):
    print(i)
\end{code}
\end{document}
```

Code 9: Code LaTeX avec ajout de code source Python

Le code est affiché avec une numérotation des lignes et surligné grâce à highlight.js.

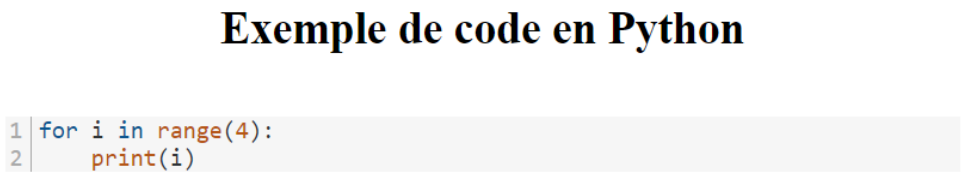


Fig. 4 : Rendu de la page HTML avec du code source

Actuellement, seul le langage Python peut être exécuté dans le navigateur. Pour cela, il faut ajouter le paramètre optionnel "interactive".

```
\documentclass[a4paper,11pt]{article}
\usepackage{commonpackages}

```

```latex
\begin{document}
\title{Exemple de code en Python}
\date{}
\maketitle
\begin{code}[interactive]{python}
for i in range(4):
    print(i)
\end{code}
\end{document}
```

Code 10: Code LaTeX avec ajout de code source Python exécutable

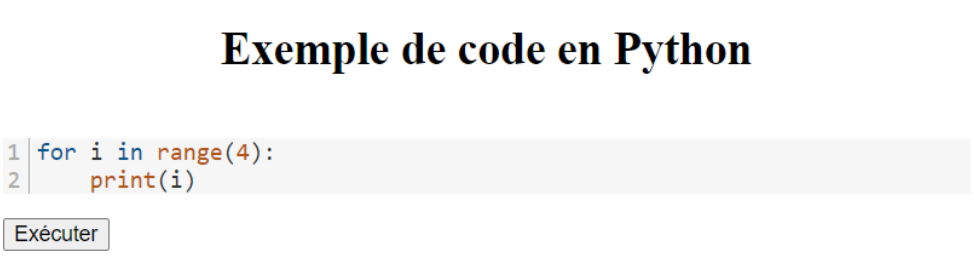


Fig. 5 : Rendu de la page HTML du code qui pourra être exécuté

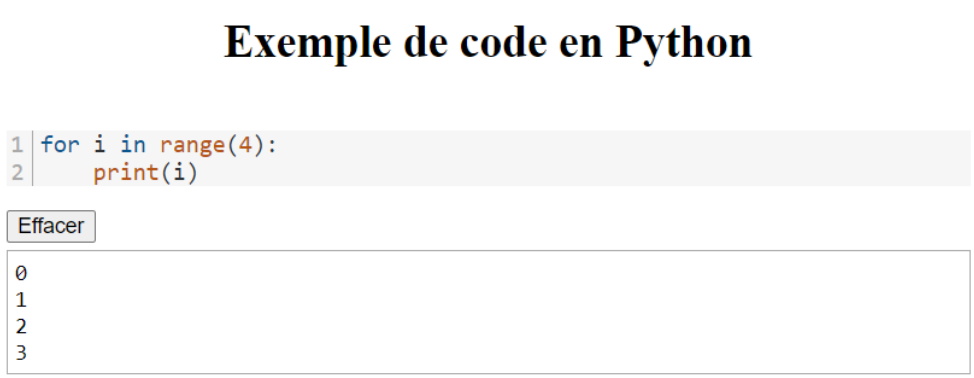


Fig. 6 : Rendu de la page HTML du code après exécution

L'affichage de code contenant le module Turtle est fait dans un canevas graphique, il est donc nécessaire de spécifier, quand il est utilisé. Cela se fait au moyen du paramètre optionnel "interactive_turtle".

```latex
\documentclass[a4paper,11pt]{article}
\usepackage{commonpackages}

\begin{document}
\title{Exemple de code en Python avec le module Turtle}
\date{}
\maketitle
\begin{code}[interactive_turtle]{python}
```

```
from turtle import *
from random import randint

penup()
goto(-100, -100)
pendown()
nb_cote_max = int(input("Combien de côtés voulez-vous au maximum? "))
colormode(255)
pensize(3)
for i in range(3, nb_cote_max):
    for j in range(i):
        forward(100)
        left(360/i)
    pencolor(randint(0,255), randint(0, 255), randint(0,255))
\end{code}
\end{document}
```

Code 11: Code LaTeX avec ajout de code source Python exécutable avec le module Turtle

Exemple de code en Python avec le module Turtle

```
from turtle import *
from random import randint

penup()
goto(-100, -100)
pendown()
nb_cote_max = int(input("Combien de côtés voulez-vous au maximum? "))
colormode(255)
pensize(3)
for i in range(3, nb_cote_max):
    for j in range(i):
        forward(100)
        left(360/i)
    pencolor(randint(0,255), randint(0, 255), randint(0,255))
```

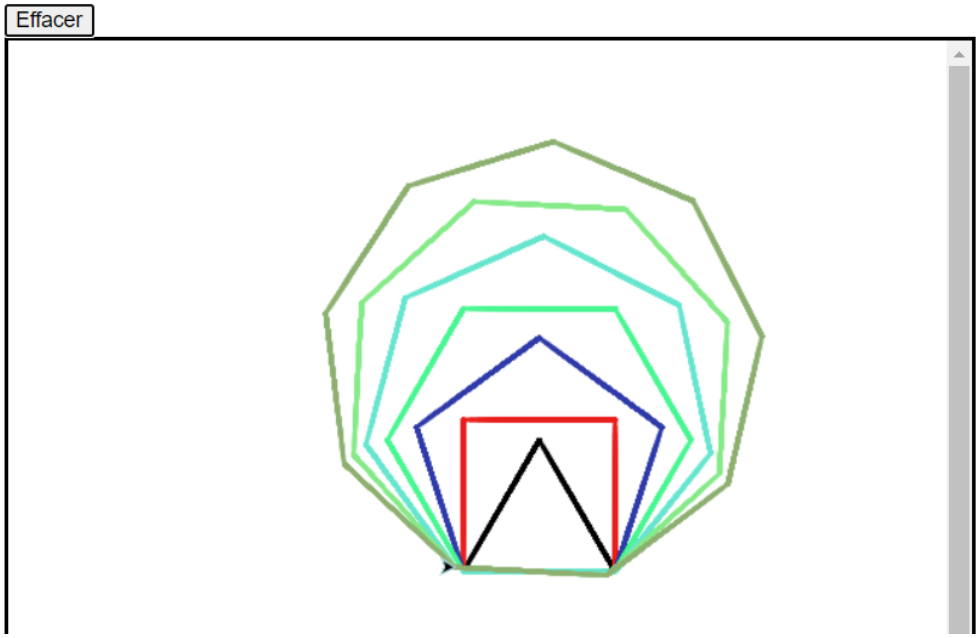


Fig. 7 : Rendu de la page HTML du code contenant le module Turtle après exécution

3.4.5 Solution

Afin de pouvoir afficher ou non la solution des exercices, celle-ci sera inclue dans un environnement solution, c'est-à-dire qu'elle sera délimitée par les balises \begin{solution} et \end{solution}.

```
\documentclass[a4paper,11pt]{article}
\usepackage{commonpackages}

\begin{document}
\title{Produits remarquables - Exercice}
\date{}
\maketitle

Calculer à l'aide des produits remarquables.
```

```
\begin{multicols}{2}
\begin{enumerate}
\item $(x-y)^2=$
\item $(x-1)^2=$
\item $(2c+1)(2c-1)=$
\item $(x-2)(x+2)=$
\end{enumerate}
\end{multicols}

\begin{solution}
\begin{enumerate}
\item $x^2-2xy+y^2$
\item $x^2-2x+1$
\item $4c^2-1$
\item $x^2-4$
\end{enumerate}
\end{solution}
\end{document}
```

Code 12: Code LaTeX avec ajout de solution à un exercice

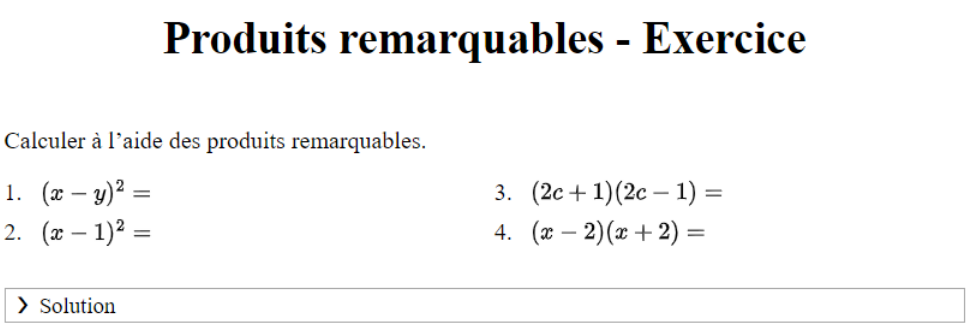
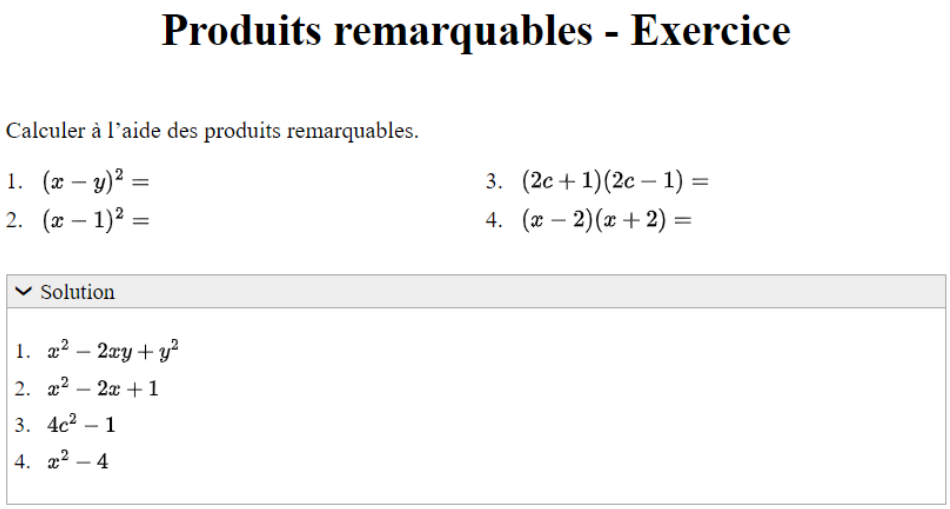


Fig. 8 : Rendu de la page HTML avec solution cachée



Produits remarquables - Exercice

Calculer à l'aide des produits remarquables.

1. $(x-y)^2 =$
2. $(x-1)^2 =$
3. $(2c+1)(2c-1) =$
4. $(x-2)(x+2) =$

Solution

1. $x^2 - 2xy + y^2$
2. $x^2 - 2x + 1$
3. $4c^2 - 1$
4. $x^2 - 4$

Fig. 9 : Rendu de la page HTML avec solution affichée

Il est tout à fait possible d'ajouter du code exécutable dans les solutions.

3.5 Exemples

Des exemples de page qui montrent l'utilisation des différentes fonctionnalitsés, sont mis à disposition sur le site suivant : http://t-doc.org/

4 Alternatives considérées

Tout au long du projet, des décisions ont été prises, dont certaines, bien qu'apparemment judicieuses au départ, ont conduit à une impasse. Face à des problèmes insolubles, il a fallu renoncer à certaines approches, malgré les nombreuses heures déjà investies dans leur développement.

4.1 Format du document d'origine

Afin de faire le maximum du rendu côté client, il semblait logique d'utiliser le HTML comme format d'origine. Il a l'avantage de pouvoir directement intégrer des images, des vidéos et des fenêtres Geogebra. Il restait donc à trouver une solution pour le contenu mathématique. Écrire des formules mathématiques en HTML peut être compliqué, prend du temps et l'affichage n'est pas toujours acceptable. De plus, en gardant en tête l'idée de pouvoir réutiliser des documents déjà crées, il faut donc pouvoir intégrer du LaTeX dans le HTML. Le choix suivant a été fait : le document complet est écrit en HTML, à l'exception des parties de contenu mathématique qui sont rédigées en LaTeX. Celles-ci seront délimitées par des balises HTML personnalisées <latex> et </latex>. La conversion se fera dans l'idéal directement par le navigateur, si ce n'est pas possible, un serveur pourrait répondre à des sous-requêtes afin de faire les transformations nécessaires.

4.2 Transformation de LaTeX en HTML

Il faut donc trouver un outil qui permet de transformer des bouts de LaTeX en HTML, directement dans le navigateur. Latex.js[18] est un traducteur de LaTeX en HTML qui répond à ces critères.

Voici ses principales caractéristiques :

- Il est open source et développé par Michael Brade.
- Il est écrit en JavaScript et s'exécute directement dans le navigateur, ainsi aucune dépendance externe ne doit être chargée.
- Il peut être intégré sur un site Web à l'aide du composant Web fourni.
- Il permet de modifier la DOM[19] du document HTML.
- Il peut être utilisé pour des documents LaTeX complets ou des fragments de LaTeX, c'est-à-dire sans préambule.
- Il est composé d'un parseur et d'un générateur, il serait donc possible de générer autre chose que du HTML en sortie.
- Le rendu HTML est très proche du rendu attendu avec LaTeX, cela grâce au CSS.
- De nouvelles macros peuvent être ajoutées en JavaScript.
- Le rendu est rapide, car LaTeX.js n'a pas besoin de parser le document plusieurs fois.

En testant cet outil avec des documents de cours, je constate que plusieurs fonctionnalités essentielles de LaTeX ne sont pas supportées[20], par exemple les images inclues directement dans le LaTeX ne s'affichent pas. Cela signifie qu'il faut le faire manuellement en HTML. De plus, certains symboles mathématiques comme ≠ n'apparaissent pas correctement. Le problème semble provenir du fait que KaTeX, une bibliothèque de composition mathématique pour le Web, n'a pas été mise à jour. Malheureusement, ce projet, comme beaucoup d'autres, n'est plus développé ou mis à jour depuis 2021. N'ayant pas la possibilité de résoudre ce problème par moi-même dans un temps raisonnable, l'utilisation de cet outil a été abandonnée.

Une autre solution intéressante est MathJax[21], un moteur d'affichage JavaScript open source pour les notations LaTeX. Il permet d'écrire du LaTeX directement dans la partie <body> du HTML. Mais, dans ce cas, il ne prend en charge qu'une partie des commandes LaTeX, c'est-à-dire l'écriture mathématique, les formules ou les symboles, mais pas les tableaux ou les listes écrites en LaTeX.

18. https://latex.js.org/
19. Document Object Model - représentation de la page HTML (structure et contenu) par un arbre
20. voir Annexe 1 : Tableau comparatif de LaTeX.js et make4ht
21. https://docs.mathjax.org/en/latest/

Cela signifie que la réutilisation de documents déjà crées est difficile. De plus, le fait d'écrire directement dans le <body> pose certains problèmes comme l'utilisation des signes mathématiques < ou >.

4.3 Sous-requêtes

Pour éviter les problèmes cités ci-dessus, il est possible d'utiliser des composants Web MathJax. Ainsi le navigateur sait, à quelles parties de la page HTML il doit appliquer MathJax, et de quelle manière. Mais la syntaxe relativement compliquée ne peut pas être utilisée directement par un utilisateur lambda, il faut donc automatiser ce processus.

Fonctionnement des sous-requête en utilisant MathJax :

1. L'utilisateur place le contenu LaTeX dans la balise HTML personnalisée <latex> </latex>.
2. Lorsque le navigateur parcourt le document HTML et rencontre une telle balise, il envoie une sous-requête au serveur avec le contenu LaTeX.
3. Le serveur crée un composant Web MathJax et y place le contenu LaTeX, ensuite il le renvoie au navigateur.
4. MathJax affiche correctement le contenu du composant Web MathJax.

Cette solution permet de paralléliser les rendus et donc d'optimiser la vitesse du rendu. Malheureusement lorsqu'il y a plusieurs composants Web, MathJax n'affiche correctement qu'un seul des composants et pas les autres. L'utilisation de MathJax avec des sous-requêtes a donc été abandonnée.

5 Problèmes rencontrés

Il existe sur le Web énormément d'outils développés par des particuliers et mis à disposition, notamment sur GitHub. Comme l'idée principale de ce projet était d'utiliser au maximum des outils existants, il a fallu faire de nombreuses recherches, sélectionner ceux qui semblaient adéquats, comprendre comment ils fonctionnent et les intégrer correctement. Lors de ce processus, les problèmes suivants ont été rencontrés :

- Arrêt du développement :
  Beaucoup d'outils sont développés pour un but très précis et lorsque ce but est atteint, ils sont laissés à l'abandon. En testant LaTeX.js, un problème d'affichage de certains symboles mathématiques est apparu. L'origine du bug semble connue : la librairie KaTeX, inclue dans le projet, n'a pas été mise à jour. Malheureusement, malgré le signalement de plusieurs

utilisateurs, rien n'a été fait. Il aurait été possible de reprendre le développement de cet outil de mon côté et de corriger ce problème, moyennant beaucoup de travail. Mais comme LaTeX.js ne répondait pas à tous les critères de mon projet, cette idée a été abandonnée.

- "Mauvais choix" :
  Dans un tel projet, il faut constamment faire des choix entre différents outils ou approches. Une idée qui semble très bonne à un moment donné du processus, devient parfois un mauvais choix par la suite, car mène dans une impasse ou est une solution trop compliquée. Du coup, il faut revenir en arrière et tout le travail fourni est "perdu". Cela a été le cas notamment avec l'utilisation de LaTeX.js et les composants Web MathJax. Heureusement, quelques connaissances acquises ont pu être réutilisées.

- Manque de documentation :
  Le package make4ht qui fait exactement ce qu'il faut pour ce projet, est très peu documenté. Le fonctionnement du fichier de configuration n'est pas détaillé et il n'existe que très peu d'exemples sur le Web.

- Messages d'erreur de make4ht :
  De manière générale, les messages d'erreur de LaTeX sont peu transparents. Mais comme la ligne où se trouve l'erreur est indiquée, il est possible de résoudre le problème. Malheureusement les messages d'erreur de make4ht, à l'exception des erreurs de syntaxe dans le document d'origine en LaTeX, sont souvent incompréhensibles. Il faut donc procéder par tests successifs pour comprendre ce qui se passe et comment corriger la situation.

- Compatibilité de LaTeX :
  LaTeX est un langage qui permet de faire du formatage de nombreuses manières différentes, mais cela le rend très complexe et difficile à rendre compatible avec toutes les fonctionnalités. Make4ht fait appel à différents outils lors du processus de conversion, notamment tex4ht. Lors de l'installation des packages texlive[22] pour la création du conteneur Docker, t-doc ne fonctionnait plus. Il a été compliqué de comprendre pourquoi. Les problèmes suivants ont été découverts :

  - en utilisant l'image alpine linux[23], il manquait le programme tex4ht qui rendait impossible la génération des documents,
  - en utilisant l'image Ubuntu, la version de make4ht n'était pas la bonne et renvoyait une erreur de dépassement de capacité.

22. https://www.tug.org/texlive/
23. https://www.alpinelinux.org/

6 Suite du projet

Le projet dans sa version actuelle est utilisable, mais il reste de nombreuses fonctionnalités à développer.
Améliorations pour l'élève :

- pouvoir cliquer sur les images et les agrandir,
- ajouter un bouton stop, afin de pouvoir arrêter l'exécution du code lors de boucles infinies,
- pouvoir éditer du code source directement dans le navigateur et ensuite l'exécuter,
- ajouter un environnement de test sous forme de bac à sable,
- pouvoir résoudre et valider des exercices de maths et de programmation directement en ligne,
- pouvoir se connecter à une session pour enregistrer la progression.

Améliorations pour l'enseignant :

- pouvoir ajouter des vidéos personnelles stockées sur le serveur,
- pouvoir ajouter des documents Geogebra stockés sur le serveur,
- pouvoir exécuter, dans le navigateur, d'autres langages de programmation que Python,
- utiliser la génération automatique d'exercices en LaTeX pour créer facilement des exercices de drill.
- avoir des session pour les élèves afin de pouvoir suivre leur progression,
- sécuriser le rendu LaTeX → HTML afin de pouvoir déployer t-doc de manière centralisée, par exemple pour plusieurs enseignants d'un établissement,
- pouvoir créer et éditer les documents directement en ligne :
  - dans le navigateur, deux zones : une pour éditer le document LaTeX et l'autre pour prévisualiser la page HTML,
  - affichage en live de la prévisualisation,
  - fonctionnalités standards : ouvrir, sauvegarder au format d'origine ou au format HTML,
  - stockage des documents en ligne
  - ajout d'images, de vidéos, de liens, … avec du "drag and drop".

Conclusion

T-doc a pour but de proposer, aux enseignants de mathématiques ou d'informatique au collège, un outil qui permet de combiner du LaTeX, du code Python, des images, des vidéos et des documents Geogebra pour créer des séquences d'enseignement en autonomie pour les élèves. Les élèves peuvent, en plus du polycopié en format PDF, profiter de pages Web pour revoir une notion vue en classe, réviser pour un examen ou découvrir une nouvelle notion.

Dans un premier temps, le HTML a été choisi comme format d'origine du document. Ainsi, les vidéos, les images, les fenêtres Geogebra peuvent être facilement intégrées, car cela est supporté par le HTML. La conversion de LaTeX s'est faite, grâce à LaTeX.js, directement dans le navigateur. Malgré une utilisation facile et un résultat intéressant, cet outil a été abandonné, car il ne permet pas de convertir toutes les fonctionnalités LaTeX souhaitées.

Afin de gérer le maximum du rendu côté client, une deuxième approche a été de mettre le contenu LaTeX dans des balises HTML personnalisées <latex> </latex> et de demander à MathJax de gérer le rendu. Cela s'est fait sous la forme de sous-requêtes envoyées au serveur pour chaque balise <latex> rencontrée. Le navigateur envoie le contenu LaTeX au serveur et le serveur crée un composant Web MathJax, y place le contenu LaTeX et renvoie le tout au navigateur qui, grâce à MathJax, affiche le contenu mathématique correctement. Comme l'affichage simultané de plusieurs composants n'est pas possible, cette solution n'était pas acceptable.

L'utilisation de sous-requêtes n'étant pas possible, l'idée d'utiliser du HTML comme format d'origine n'a plus de sens. C'est pourquoi à partir de ce moment, le format d'origine sera LaTeX. Le document LaTeX complet est converti en une seule fois en HTML. Pour cela il existe de nombreux outils, dont la différence notoire est la qualité du rendu du contenu mathématique et la possibilité de personnaliser celui-ci. Make4ht, grâce à plusieurs options et à un fichier de configuration, permet de personnaliser le processus, notamment en choisissant MathJax pour l'affichage des formules mathématiques. Cet outil répond à tous les critères souhaités pour ce projet.

La réalisation de t-doc a été l'occasion de développer un projet de l'émergence de l'idée au déploiement sur un serveur. En plus de l'approfondissement de mes connaissances en LaTeX, Python, JavaScript, Django, …, qui me seront utiles pour mon enseignement, j'ai aussi appris à utiliser correctement un contrôle de révision, dans mon cas Mercurial, et à créer une image de mon projet pour Docker. À l'heure actuelle, le projet est fonctionnel. Et même s'il reste de nombreuses fonctionnalités à développer pour l'améliorer, je me réjouis de l'utiliser avec mes élèves.

Remerciements

Mes remerciements vont tout d'abord au Prof. Dr. Jacques Pasquier qui m'a suivi tout au long de ce projet, en me permettant de réaliser une application qui me sera utile pour mon enseignement. Je souhaite aussi remercier mon mari qui m'a encouragée, soutenue et aidée sous différentes formes : le coaching dans la gestion de projets, les nombreuses heures passées à discuter pour trouver la meilleure approche, les bonnes solutions ou de nouvelles idées et surtout les bons conseils lorsque j'étais bloquée.

Webographie

BRADE MICHAEL « JavaScript LATEX to HTML5 translator », 2017-2021. <https://latex.js.org/>.

DJANGO SOFTWARE FOUNDATION « django », 2005-2024. <https://www.djangoproject.com/>.

DOCKER INC. « docker docs », 2013-2024 <https://docs.docker.com/>.

GEOGEBRA « Applis Maths Geogebra », 2024. <https://www.geogebra.org/>.

« highlight.js », 2024. <https://highlightjs.org/>.

INTERNATIONAL GEOGEBRA INSTITUTE « Reference : GeoGebra Apps Embedding », 2024. <https://wiki.geogebra.org/en/Reference:GeoGebra_Apps_Embedding>.

MICHAL HOFTICH « The make4ht build system », 2023. <https://ctan.org/pkg/make4ht?lang=en>.

PYTHON SOFTWARE FOUNDATION « Python », 2001-2024. <https://www.python.org/>.

THE MATHJAX CONSORTIUM « MathJax Documentation », 2024. <https://docs.mathjax.org/en/latest/index.html>.

SCOTT GRAHAM « Skulpt », 2015. <https://skulpt.org/>.

Annexes

Annexe 1: Tableau comparatif de LaTeX.js et make4ht

Élements LaTeX	LaTeX.js	make4ht
Formules mathématiques $ ...$	ok	ok, avec mathjax (mathml pose problème avec certains éléments)
Formules ”hors ligne” $$ ...$$	ok	ok
Itemize (listes non numérotées)	ok	ok
Enumerate (listes numérotées)	ok, mais pas possible de modifier le format	ok
Multicolonnes	ok	ok
Array dans l'environnement math	ok	ok
Tabular	pas possible, mais pourrait être remplacé par array	ok
Compactenum	pas possible, mais pourrait être remplacé par enumerate	ok
Includegraphics (images)	pas possible directement	ok, mais il faut spécifier la largeur en fonction de la largeur du texte
Symboles latex	certains caractères n'ont pas été définis ou posent problème comme ≠	ok
Espacements verticaux	ok	ils sont ignorés
Options pour enumerate et itemize	pas possible et s'ils sont présents, posent problème	ok
Newpage	ignorés	ignorés
Array avec des espacements de colonnes définis	pas possible	pas possible

Annexe 2 : Outils de rendu LaTeX en HTML

Nom	Site	Description
Webtex	https://pkgw.github.io/webtex/	Webtex a été développé pour afficher correctement les articles scientifiques dans le navigateur. Ce projet date de 2016 et ne semble plus développé.
Pandoc	https://pandoc.org/index.html	Pandoc permet de convertir des documents numériques d'un format à un autre. Il faut l'installer séparément.
TexLive ou MikeTek ou xelatex		Les distributions de LaTeX incluent en général un exécutable pour convertir du LaTeX en HTML en ligne de commande. Cet exécutable convertit le document entier en un fichier HTML et un fichier CSS pour la mise en page.
MiniLatex	https://minilatex.lamdera.app/	MiniLatex permet de convertir du LaTeX en HTML directement en ligne. Il permet aussi l'hébergement direct des pages. Comme c'est une application à part entière, il n'est pas possible de la personnaliser.
latex2html	https://github.com/latex2html/latex2html/	Latex2html permet de convertir du LaTeX en HTML. Les formules mathématiques sont converties en images (.gif).
make4ht	https://ctan.org/pkg/make4ht?lang=en	make4ht est un système de construction simple pour tex4ht, un convertisseur de TEX en HTML. Il permet de personnaliser la conversion.
latex-to-html	https://lib.rs/crates/latex-to-html	latex-to-html permet de convertir du LaTeX en HTML. Il transforme les formules en images vectorielles.