La programmation en simple

Chapitre 1 : L’introduction

La programmation informatique, c’est un peu comme découvrir un nouveau pays avec une nouvelle langue. Au début, vous aurez envie de tout savoir sur absolument tout, mais c’est impossible. Est-ce que vous êtes d’accord avec cela ? Maintenant, imaginez que la programmation, c’est aussi prendre plaisir à apprendre continuellement, sans jamais pouvoir tout savoir, car de nouvelles technologies apparaissent sans cesse. Si vous êtes d’accord, bienvenue dans l’univers de ce livre. Sinon, merci d’avoir acheté cet ouvrage, mais il ne vous sera sans doute d’aucune utilité.

Ce que vous venez de lire ? Un algorithme. C’est un concept que j’ai mis des mois à comprendre lorsque j’ai commencé à apprendre à coder. Vous ne vous en êtes peut-être pas rendu compte, mais les algorithmes sont omniprésents dans notre vie quotidienne, et ce chapitre en sera rempli.

Je ne suis pas ici pour vous apprendre à coder, car cela serait inutile pour vous comme pour moi. Je ne suis pas professeur, et par conséquent, vous n’êtes pas mes élèves. Ce livre va simplement vous aider à acquérir quelques principes fondamentaux de la programmation. Je vais vous fournir des exemples, des situations, etc.

Avez-vous compris ce qu’est un algorithme ou êtes-vous encore en train de réfléchir à cela ? Si oui, je vais vous donner une explication ; si non, vous pouvez passer au paragraphe

suivant. Un algorithme est une sorte de question à double tranchant pour les plus simples, ou plusieurs tranchants pour les plus complexes. C'est comme suivre une recette de cuisine. Prenons l'exemple des crêpes : la recette de base est œuf, lait et farine. Si vous n'avez pas de sucre ou de beurre, vous pouvez soit remplacer ces ingrédients, soit vous en passer. L'objectif d'un algorithme est d'étudier chaque possibilité sous forme de schéma dans la vie de tous les jours, ou sous forme de code. Lorsque vous commencez à coder, vous pourriez même envisager de créer une application pour faire des crêpes. Pourquoi pas ?

Ici, pas de leçon, vous êtes libre de lire mon livre ou non, de suivre mes conseils ou non. En fin de compte, je considère mon livre comme un algorithme : c'est à vous de décider ce que vous souhaitez suivre ou non.

Le message que je veux faire passer dans cet ouvrage ? Ma passion pour l'informatique, de manière générale, et ma passion pour la programmation, de manière plus spécifique.

Laissez-moi vous raconter comment j'ai découvert le code et pourquoi j'ai voulu apprendre tout cela. J'ai toujours été immergé dans le monde de l'informatique. Pour la programmation, c'est venu plus tard, vers l'âge de 16 ans, lorsque j'ai eu l'idée de créer un site web avec un ami pour vendre des liquides de cigarette électronique. En à peine deux mois de code, il m'est devenu évident que je ne pouvais pas créer quelque chose de propre.

Durant cette période, j'ai été hospitalisé quelques jours et n'avais que mon téléphone avec moi. J'ai passé tout mon temps

à penser à mon code et à mon ordinateur, comme si c'était la chose la plus importante. J'en rêvais même la nuit.

À mon retour, la première chose que j'ai faite a été d'allumer mon ordinateur pour continuer mon code. Je voulais que ce soit parfait. Lorsque j'ai vu le résultat final, j'étais tellement déçu que j'ai tout effacé pour tout recommencer. J'ai recommencé encore et encore jusqu'à ce que le résultat me convienne. Bien sûr, il y avait plein de choses que je n'avais pas prises en compte, comme la nécessité d'une interface d'inscription et de connexion. Finalement, nous avons opté pour une boutique en ligne toute faite pour réaliser notre idée de départ.

Ce que je veux vous faire comprendre, c'est que vous ne pourrez pas réaliser toutes vos idées dès le départ. Vous devrez passer par plusieurs étapes : « Je ne sais pas faire cela, soit j'essaie encore et encore, soit je prends une solution plus simple, soit je fais appel à quelqu'un de plus expérimenté que moi. » Encore un algorithme.

En somme, il faut persévérer, ne jamais abandonner et ne pas hésiter à demander de l'aide. Ce sont les trois clés pour avancer dans ce milieu. Même si votre premier projet semble loin d'être adapté à votre niveau, notez cette idée quelque part et cherchez des projets qui soient réalisables pour vous. Je sais que c'est frustrant, mais dites-vous que vous serez capable de réaliser ce premier projet dans quelques mois. Préférez-vous attendre quelques mois, continuer d'apprendre et d'évoluer, ou foncer tête baissée dans un projet qui n'aboutira pas ? À vous de choisir. Quelle que soit votre décision, cela déterminera sans doute votre avenir dans ce domaine.

Je tiens à préciser que tout ce que j’écris ici reste des conseils, et c’est ce que j’aurais aimé savoir lorsque j’ai décidé de me lancer dans la programmation.

Chapitre 2 : Les différents langages de programmation

Maintenant que vous savez à quoi vous attendre, passons aux choses sérieuses. Nous allons parler des différents langages de programmation. Oui, il en existe une multitude. Le problème lorsque l'on débute, c'est qu'il ne faut pas avoir peur d'apprendre. Imaginez cela comme un nouveau pays avec plusieurs nouvelles langues. Vous vous sentirez forcément perdu au début.

Pourquoi seulement trois nouveaux pays alors qu'il y a plein de nouvelles langues ? Laissez-moi vous expliquer. Les trois pays correspondent aux trois interfaces de développement web : le front-end, le back-end et le full stack. Ces trois pays parlent chacun différentes langues, à l'exception du full stack qui parle toutes les langues possibles.

- **Front-end** : HTML / CSS / JavaScript
- **Back-end** : Java / Python / PHP / C / C++, etc.
- **Full-stack** : le mélange des deux.

Rassurez-vous, pour le back-end, vous n'êtes pas obligé de tout apprendre. Un ou deux langages suffisent. En revanche, pour le front-end, vous devrez apprendre les trois.

Voici à quoi correspondent ces trois pays :

- **Front-end** : pour créer des sites web et styliser des applications.
- **Back-end** : pour gérer la logique d'une application ou d'un site web.
- **Full-stack** : la combinaison des deux.

Personnellement, je vous conseille de commencer par le front-end, car les bases seront plus faciles à assimiler et vous pourrez déjà réaliser pas mal de choses. Après, le choix dépend de votre goût pour la logique : si vous aimez cela, alors commencez par des langages back-end.

Je vous recommande de connaître au moins les bases de chaque langage et à quoi ils servent, cela vous sera toujours utile. Vous trouverez un index à la fin du livre qui vous donnera des informations sur ce que chaque langage permet de faire, ainsi que ses avantages et inconvénients.

Chapitre 3 : Les éditeurs de code

Un éditeur de code, c'est comme un éditeur de texte, mais vous n'y retrouverez rien en commun, à part la possibilité d'écrire. Vous aurez besoin d'un éditeur de code pour commencer à coder. Selon le langage que vous choisissez d'étudier, vous sélectionnerez celui qui convient le mieux à vos besoins. Je vous conseille de commencer par un éditeur simple, puis de passer à des logiciels plus complexes lorsque vous aurez acquis quelques bases. Cela vous évitera d'être perdu dès le départ et vous permettra de découvrir plusieurs outils.

Ces éditeurs sont généralement faciles à utiliser, surtout si vous maîtrisez l'anglais. Si ce n'est pas le cas, pas de panique, cela viendra avec le temps. Sachez que le code informatique est principalement écrit en anglais, peu importe le langage que vous apprendrez.

Tout le monde galère au début, mais je vous assure que cela en vaut la peine. Le monde de la programmation est fascinant et technique. Si vous n'êtes pas déjà à l'aise avec la technologie, cela sera compliqué, mais pas impossible.

Pour vous aider, je fournirai une liste d'éditeurs de code à la fin du livre, avec des explications sur comment les télécharger et commencer à les utiliser. Vous constaterez que beaucoup d'entre eux sont très simples d'utilisation et qu'il n'est pas nécessaire de passer des heures à les maîtriser.

Certaines applications d’édition de code sont spécialisées pour un langage particulier. Par exemple, BlueJ est exclusivement utilisé pour coder en Java, mais il est très pratique. Pour ma part, j’utilise Visual Studio Code, qui est polyvalent et adapté à n’importe quel langage. C’est un excellent choix pour débuter.

Personnalisez votre éditeur de code à votre goût. Vous pouvez commencer à coder dès que votre environnement de travail est prêt, c’est-à-dire lorsque vous avez un bureau calme, un ordinateur prêt à recevoir votre code, et une idée en tête pour votre premier programme. Vous pouvez également suivre des cours en ligne, des vidéos ou lire des livres, mais assurez-vous de rester concentré.

Chapitre 4 : Les bonnes pratiques pour coder

Une fois tout cela fait, vous avez besoin de quelques astuces pour coder le plus proprement possible, minimiser les erreurs dans votre code et apprendre à travailler en pensant à l'avenir.

Pour coder proprement, rien de plus simple : il suffit de bien indenter votre code. Vous verrez que lorsque vous travaillerez avec des langages de back-end, vous n'aurez d'autre choix que d'indenter correctement votre code, sous peine d'erreurs.

Concernant la réduction des erreurs, sachez que même le meilleur développeur a déjà commis des erreurs dans son code. C'est tout à fait normal, surtout au début. Vous risquez d'en faire beaucoup, et pour en limiter le nombre, il est essentiel d'apprendre à bien indenter votre code. Ensuite, relisez-le et commentez-le ! Commenter son code est une excellente pratique, car cela vous permettra, dans le futur, de savoir ce que vous avez fait et comment vous y avez réfléchi. De plus, lorsque vous travaillerez avec d'autres sur le même projet, il est important de comprendre qui a fait quoi, pourquoi, et comment chaque personne a réfléchi à son code.

Au début, vous aurez envie d'apprendre tout par cœur, mais ce n'est pas la meilleure approche. Il vaut mieux écrire un code réfléchi et le réviser plusieurs fois, plutôt qu'un code écrit bêtement grâce à des choses mémorisées. Bien sûr, il y a des

éléments à apprendre par cœur, comme les balises en HTML ou le fonctionnement de chaque langage. Cependant, il n'est pas nécessaire d'apprendre toute la logique des langages par cœur ; cela viendra naturellement avec la pratique.

Il est normal, sur certains projets, de se sentir frustré à cause des bugs ou du manque de compétences pour réaliser ce que vous souhaitez. Comprenez que le code ne s’apprend pas en quelques mois. Tout au long de votre parcours, vous continuerez à apprendre sans jamais vous reposer sur vos acquis. De nouvelles technologies apparaissent en permanence et elles vous seront un jour utiles, alors ne les négligez pas. Concernant les versions des langages de programmation, bien qu’une version puisse rester actuelle pendant plusieurs années, elle finira par être dépassée. Si vous continuez à apprendre régulièrement, vous serez mieux préparé et plus apte à vous adapter.

Quand vous codez, pensez à prendre des pauses. Sortez prendre l’air, allez voir votre famille, faites autre chose, puis revenez à votre code lorsque vous vous sentez prêt. Lorsque vous décidez de vous lancer dans un nouveau projet pour lequel vous n’avez pas encore toutes les compétences requises, habituez-vous à avoir un ou deux autres projets en cours que vous connaissez bien. Cela vous évitera des crises de nerfs devant votre ordinateur. Ce que je vous dis peut sembler évident, mais avec le temps, vous comprendrez que cela peut vraiment vous aider. Je l’ai vécu, avec des codes remplis de bugs que j’essayais de corriger, mais qui semblaient en rajouter encore plus.

Encore une fois, le chemin que vous emprunterez dans le domaine de la programmation n’est pas simple, mais c’est un monde où la seule limite est votre imagination.

Chapitre 5 : La seule limite est votre propre imagination

Comme je vous le disais à la fin du chapitre précédent, la programmation est un monde où la seule limite est votre imagination. Cela signifie qu'il n’y a aucune limite à la création, si ce n’est celle de votre imagination. Vous pouvez tout construire, même l’impensable. Cependant, sans l’imagination nécessaire pour concevoir votre projet, rien d’extraordinaire ne pourra se produire.

Si vous avez plein d’idées de projets, mais qu'elles ne sont pas assez solides, puissantes ou personnelles, il est peu probable qu'elles aboutissent. Avant de vous lancer dans un gros projet, réfléchissez à ces trois questions :

- Qu’est-ce que mon projet apporte de plus à Internet ?
- Qu’est-ce que mon projet m’apporte personnellement (en dehors de la fierté) ?
- Qu’est-ce que je peux améliorer par la suite dans mon projet ? (Il est important de penser à l’avenir.)

Une fois que vous aurez répondu à ces trois questions et que vous aurez peaufiné votre projet, vous serez prêt à vous lancer dans le développement.

Lorsque vous aurez réussi à allier votre imagination et vos connaissances, votre projet sera prêt à être lancé. Je vous

conseille également de mettre votre projet sur papier ; cela vous aidera à visualiser où vous en êtes et ce qu'il vous reste à faire, ce qui est particulièrement utile pour un projet complexe ou pour un type de programmation que vous ne maîtrisez pas encore.

N’hésitez pas à faire des essais, à recommencer et à prendre votre temps pour que votre code fonctionne correctement. Pensez à toutes les éventualités.

Et surtout, aimez coder ! C’est le plus important : c’est une passion avant d’être un travail. Profitez de ce que vous faites et appréciez chaque moment !

Index

Langages de programmation :

Front-end :

- **HTML** : Permet de structurer des pages web.
- **CSS** : Sert à styliser les pages HTML.
- **JavaScript** : Dynamise les sites web.

Back-end : (À vous de faire vos recherches sur les langages de back-end, tels que Node.js, Python, PHP, Ruby, etc.)

Éditeurs de code :

- **Visual Studio Code**
- **PyCharm**
- **Notepad++**
- **BlueJ**