

# Le livre de la Programmation

---

## Les deux aspects

---

Sous la surface de la machine, le programme bouge. Sans effort, il se dilate et se contracte. De façon harmonieuse, les électrons s'éparpillent et se regroupent. Les formes affichées sur le moniteur ne sont que des vagues de surface. L'essence reste invisible, en dessous.

Lorsque les créateurs ont construit la machine, ils y ont placé le processeur et la mémoire. De ceux-ci découlent les deux aspects du programme.

L'aspect du processeur est la substance active. Il est appelé Contrôle. L'aspect de la mémoire est la substance passive. Il est appelé Données.

Les Données sont constituées de simples bits, mais elles prennent des formulations complexes. Le Contrôle n'est constitué que d'instructions simples, mais il accomplit des tâches difficiles. De ce qui est petit et insignifiant naissent des choses grandes et complexes.

Les Données constituent la source du programme. Le Contrôle en découle. Le Contrôle procède à la création de nouvelles Données. L'un naît de l'autre, l'autre est inutile sans l'un. C'est le cycle harmonieux des Données et du Contrôle.

En soi, les Données et le Contrôle sont sans structure. Les programmeurs d'autrefois façonnaient leurs programmes à partir de cette substance brute. Au fil du temps, les Données sans formes se sont cristallisées en types de données, et le Contrôle chaotique a été restreint en structures de contrôle et en fonctions.

## Réflexions courtes

---

Lorsqu'un étudiant a demandé à Fu-Tzu quelle était la nature du cycle des données et du contrôle, Fu-Tzu a répondu : « Pensez à un compilateur qui se compile lui-même ».

Un étudiant a demandé « Les programmeurs d'autrefois n'utilisaient que des machines simples et aucun langage de programmation, et pourtant ils faisaient de beaux programmes. Pourquoi utilisons-nous des machines compliquées et des langages de programmation ? ». Fu-Tzu répondit « Les bâtisseurs d'autrefois n'utilisaient que des bâtons et de l'argile, et pourtant, ils ont fait de belles huttes ».

Un ermite passa dix ans à écrire un programme. « Mon programme peut calculer le mouvement des étoiles sur un ordinateur 286 fonctionnant sous MS DOS », annonce-t-il fièrement. « Plus personne ne possède un ordinateur 286 ou n'utilise MS DOS », répondit Fu-Tzu.

Fu-Tzu avait écrit un petit programme qui était plein d'états globaux et de raccourcis douteux. En le lisant, un étudiant demanda : « Vous nous avez mis en garde contre ces techniques, et pourtant je les trouve dans votre programme. Comment est-ce possible ? » Fu-Tzu a dit « Il n'y a pas besoin d'aller chercher un tuyau d'arrosage quand la maison n'est pas en feu » <sup>1</sup>.

## Sagesse

---

Un élève se plaignait des nombres numériques. « Quand je prends la racine de deux et que je la remets au carré, le résultat est déjà inexact ! ». En l'entendant, Fu-Tzu s'est mis à rire. « Voici une feuille de papier. Écris-moi la valeur précise de la racine carrée de deux. »

Fu-Tzu dit : « Quand on coupe contre le grain du bois, il faut beaucoup de force. Lorsque vous programmez contre le grain d'un problème, beaucoup de code est nécessaire. »

Tzu-li et Tzu-ssu se vantaient de la taille de leurs derniers programmes. « Deux cent mille lignes », dit Tzu-li, « sans compter les commentaires ! ». « Psah », dit Tzu-ssu, « le mien fait déjà presque un *million* de lignes ». Fu-Tzu dit « Mon meilleur programme a cinq cents lignes ». En entendant cela, Tzu-li et Tzu-ssu furent éclairés.

Un étudiant était assis immobile derrière son ordinateur depuis des heures, les sourcils froncés. Il essayait d'écrire une belle solution à un problème difficile, mais ne parvenait pas à trouver la bonne approche. Fu-Tzu le frappa à l'arrière de la tête et lui cria « *Tape quelque chose !* ». L'étudiant a commencé à écrire une solution laide. Après avoir terminé, il a soudainement compris la belle solution.

## Progression

---

Un programmeur débutant écrit ses programmes comme une fourmi construit sa colline, un grain de sable à la fois, sans penser à la structure globale. Ses programmes seront comme du sable meuble. Ils peuvent tenir pendant un certain temps, mais en devenant trop gros ils s'écroulent <sup>2</sup>.

Conscient de ce problème, le programmeur commencera à passer beaucoup de temps à penser à la structure. Ses programmes seront structurés de manière rigide, comme des sculptures rocheuses. Ils sont solides, mais lorsqu'ils doivent changer, il faut leur faire violence <sup>3</sup>.

Le maître programmeur sait quand appliquer la structure et quand laisser les choses dans leur forme simple. Ses programmes sont comme de l'argile, solide, mais malléable.

## Langage

---

Lorsqu'un langage de programmation est créé, on lui attribue une syntaxe et une sémantique. La syntaxe décrit la forme du programme, la sémantique décrit la fonction. Lorsque la syntaxe est belle et que la sémantique est claire, le programme sera comme un arbre majestueux. Lorsque la syntaxe est maladroite et la sémantique floue, le programme sera comme un buisson de ronces.

On a demandé à Tzu-ssu d'écrire un programme dans un langage appelé Java, qui adopte une approche très primitive des fonctions. Chaque matin, alors qu'il s'asseyait devant son ordinateur, il commençait à se plaindre. Toute la journée, il jura, accusant le langage de tout ce qui ne va pas. Fu-Tzu l'a écouté pendant un moment, puis lui a fait des reproches en disant : « Chaque langue a sa propre manière. Suis sa forme, n'essaie pas de programmer comme si tu utilisais une autre langue. »

---

# The Book of Programming

---

## The Two Aspects

---

Below the surface of the machine, the program moves. Without effort, it expands and contracts. In great harmony, electrons scatter and regroup. The forms on the monitor are but ripples on the water. The essence stays invisibly below.

When the creators built the machine, they put in the processor and the memory. From these arise the two aspects of the program.

The aspect of the processor is the active substance. It is called Control. The aspect of the memory is the passive substance. It is called Data.

Data is made of merely bits, yet it takes complex forms. Control consists only of simple instructions, yet it performs difficult tasks. From the small and trivial, the large and complex arise.

The program source is Data. Control arises from it. The Control proceeds to create new Data. The one is born from the other, the other is useless without the one. This is the harmonious cycle of Data and Control.

Of themselves, Data and Control are without structure. The programmers of old moulded their programs out of this raw substance. Over time, the amorphous Data has crystallised into data types, and the chaotic Control was restricted into control structures and functions.

## Short Sayings

---

When a student asked Fu-Tzu about the nature of the cycle of Data and Control, Fu-Tzu replied "Think of a compiler, compiling itself."

A student asked "The programmers of old used only simple machines and no programming languages, yet they made beautiful programs. Why do we use complicated machines and programming languages?". Fu-Tzu replied "The builders of old used only sticks and clay, yet they made beautiful huts."

A hermit spent ten years writing a program. "My program can compute the motion of the stars on a 286-computer running MS DOS", he proudly announced. "Nobody owns a 286-computer or uses MS DOS anymore.", Fu-Tzu responded.

Fu-Tzu had written a small program that was full of global state and dubious shortcuts. Reading it, a student asked "You warned us against these techniques, yet I find them in your program. How can this be?" Fu-Tzu said "There is no need to fetch a water hose when the house is not on fire." <sup>4</sup>

## Wisdom

---

A student was complaining about digital numbers. "When I take the root of two and then square it again, the result is already inaccurate!". Overhearing him, Fu-Tzu laughed. "Here is a sheet of paper. Write down the precise value of the square root of two for me."

Fu-Tzu said "When you cut against the grain of the wood, much strength is needed. When you program against the grain of a problem, much code is needed."

Tzu-li and Tzu-ssu were boasting about the size of their latest programs. "Two-hundred thousand lines", said Tzu-li, "not counting comments!". "Psah", said Tzu-ssu, "mine is almost a *million* lines already." Fu-Tzu said "My best program has five hundred lines." Hearing this, Tzu-li and Tzu-ssu were enlightened.

A student had been sitting motionless behind his computer for hours, frowning darkly. He was trying to write a beautiful solution to a difficult problem, but could not find the right approach. Fu-Tzu hit him on the back of his head and shouted "*Type something!*" The student started writing an ugly solution. After he had finished, he suddenly understood the beautiful solution.

## Progression

---

A beginning programmer writes his programs like an ant builds her hill, one piece at a time, without thought for the bigger structure. His programs will be like loose sand. They may stand for a while, but growing too big they fall apart. <sup>5</sup>

Realising this problem, the programmer will start to spend a lot of time thinking about structure. His programs will be rigidly structured, like rock sculptures. They are solid, but when they must change, violence must be done to them. <sup>6</sup>

The master programmer knows when to apply structure and when to leave things in their simple form. His programs are like clay, solid yet malleable.

## Language

---

When a programming language is created, it is given syntax and semantics. The syntax describes the form of the program, the semantics describe the function. When the syntax is beautiful and the semantics are clear, the program will be like a stately tree. When the syntax is clumsy and the semantics confusing, the program will be like a bramble bush.

Tzu-ssu was asked to write a program in the language called Java, which takes a very primitive approach to functions. Every morning, as he sat down in front of his computer, he started complaining. All day he cursed, blaming the language for all that went wrong. Fu-Tzu listened for a while, and then reproached him, saying "Every language has its own way. Follow its form, do not try to program as if you were using another language."

---


Tiré de : Marijn Haverbeke, The Eloquent Javascript, 1re édition, [https://eloquentjavascript.net/1st\\_edition/chapter6.html#p72845d53f6f05268](https://eloquentjavascript.net/1st_edition/chapter6.html#p72845d53f6f05268), consulté le 3 juin 2022

---

1. Ceci ne doit pas être lu comme un encouragement à la programmation bâclée, mais plutôt comme un avertissement contre l'adhésion névrotique à des règles empiriques. 

2. Se référant au danger d'incohérence interne et de structure dupliquée dans un code non organisé. 

3. Se référant au fait que la structure tend à mettre des restrictions sur l'évolution d'un programme. 

4. This is not to be read as an encouragement of sloppy programming, but rather as a warning against neurotic adherence to rules of thumb. 

5. Referring to the danger of internal inconsistency and duplicated structure in unorganised code. 

6. Referring to the fact that structure tends to put restrictions on the evolution of a program. 