

Clean code-Green code

Nouvelles Technologies de l'Information et de la Communication

Réalisé par
Mr. Mohamed Mezzi

Tuteur
Mr. Stéphane Derrode



```
require 'capybara/rspec'  
require 'capybara/rails'  
Capybara.javascript_driver = :webkit  
Category.delete_all; Category.create  
Shoulda::Matchers.configure do |config|  
  config.integrate do |with|  
    with.test_framework :rspec  
    with.library :rails  
  end  
end  
# Add additional requirements below this line  
# Requires supporting files within the same directory as this file to work.  
# run as spec/support/ and its subdirectories  
# in _spec.rb will both be loaded  
# run twice. It is recommended that you don't do this.  
with_spec.rb
```

Contenu de la Présentation

01

Le Clean Code : Une Compétence Recherchée par les Grandes Entreprises

02

L'Exemple d'Odoo

03

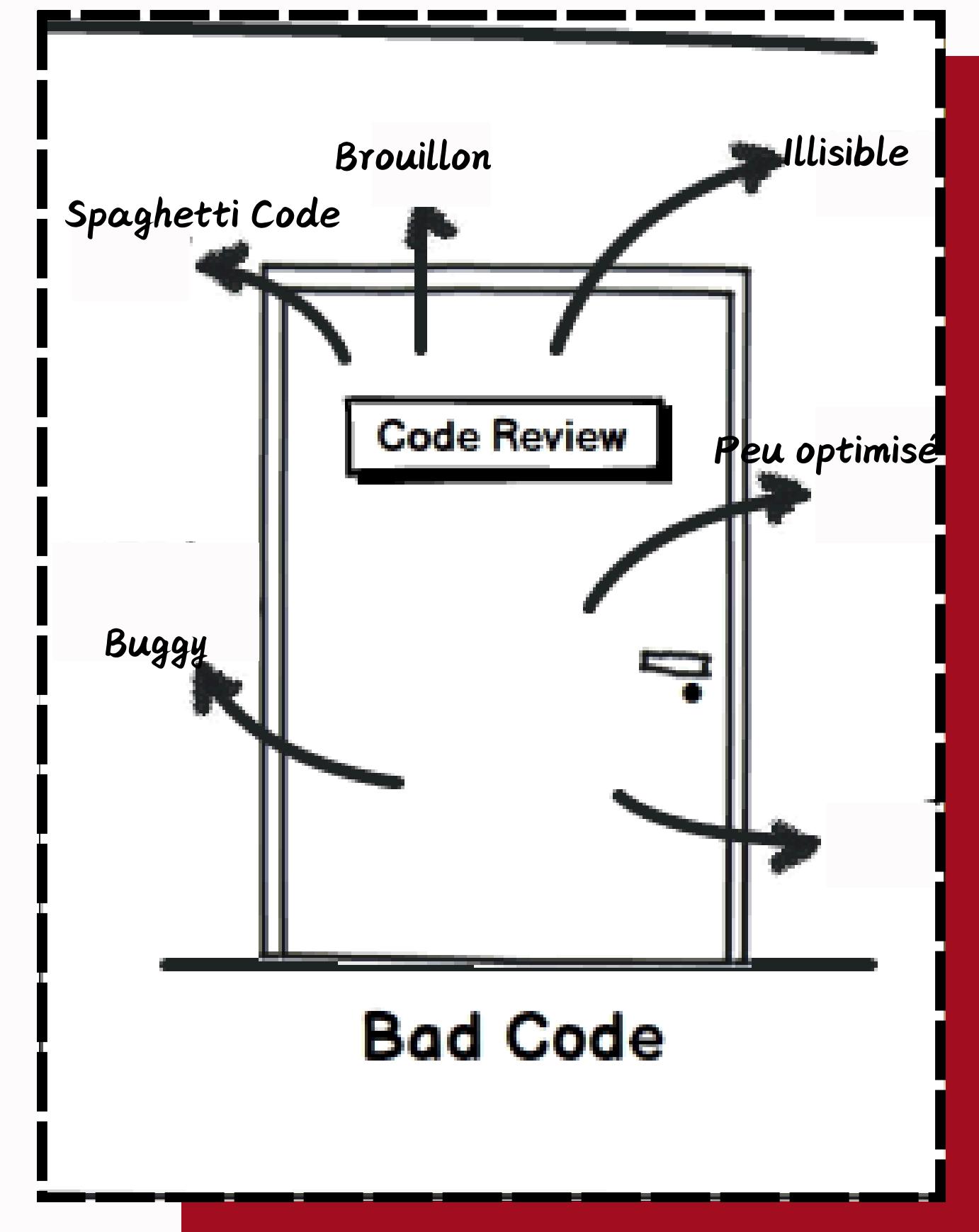
Trois Développeurs, Trois Niveaux

04

Comparaison des Langages

05

Ma Méthodologie de Veille





Le Clean Code : Une Compétence Recherchée par les Grandes Entreprises

Le Clean Code : Une Compétence Recherchée par les Grandes Entreprises

The screenshot shows a LinkedIn search interface for 'meta' in 'France'. The search bar has 'meta' entered. Below it, there are filters for 'Emplois', 'Date de publication', 'Expérience', 'Entreprise', 'À distance', 'Candidature simplifiée', and 'Tous les filtres'. The results section shows 'meta - France' with '35 résultats'. Two job listings are visible: 'Full Stack Developer – Data Structures, Algorithms' at Meta in Paris, Île-de-France, France (Sur site), and 'Client Solutions Manager' at Meta in Paris, Île-de-France, France (Sur site). The job listing for the Full Stack Developer includes a bulleted list of responsibilities. Three numbers (1, 2, 3) are overlaid on the list, with red boxes highlighting the first two items and a blue box highlighting the third item.

meta - France
35 résultats

Définir une alerte

Full Stack Developer – Data Structures, Algorithms X

Meta
Paris, Île-de-France, France (Sur site)

12 anciens élèves travaillent ici

Consulté

Client Solutions Manager X

Meta
Paris, Île-de-France, France (Sur site)

12 anciens élèves travaillent ici

Consulté

Full Stack Developer – Data Struct... Postuler ↗ Enregistrer ...

Meta · Paris, Île-de-France, France (Sur site)

Full Stack Developer – Data Structures, Algorithms Responsabilités:

1 • Strong understanding of algorithms (binary search, sorting, trees, etc.) and data structures (arrays, lists, trees).

2 • Familiarity with SQL and NoSQL databases (e.g., PostgreSQL, MongoDB).

2 • Ability to write clean, efficient, and scalable code.

2 • Strong problem-solving skills and the ability to optimize code for performance.

3 • Set KPIs and goals, design and evaluate experiments, monitor key product metrics, understand root causes of changes in metrics

3 • Experience with DevOps tools (Docker, Kubernetes) is a plus.

Minimum Qualifications:

Pourquoi les Grandes Entreprises Exigent du Clean Code ?

The screenshot shows a LinkedIn search interface for 'amazon' in 'France'. The search bar has 'amazon' and 'France' entered. Below the search bar are various filters: 'Emplois', 'Date de publication', 'Expérience', 'Entreprise', 'À distance', 'Candidature simplifiée', and 'Tous les filtres'. The main results section for 'amazon - France' shows 269 results. It lists three job postings:

- Stages 2025 – Data Engineer – Dat...** at Amazon in Clichy, Île-de-France, France. It mentions 27 former students work there and was posted 1 month ago.
- Stages 2025 – Data Engineer – Data Structures** at Amazon in Clichy, Île-de-France, France. It mentions 27 former students work there and was posted 1 month ago. This listing is marked as 'Consulté'.
- New Accounts Manager - French Speaker** at Amazon in Clichy, Île-de-France, France. It mentions 27 former students work there.

On the right side of the results, there is a detailed description for the first job posting:

Stages 2025 – Data Engineer – Dat...
Amazon · Clichy, Île-de-France, France
...
Postuler Enregistrer ...

Vos Responsabilités

- Vous travaillez directement avec nos responsables de catégories et vous les aidez à évaluer les segments de marché les plus prometteurs, à définir notre portefeuille et notre offre, à structurer le projet à lancer et à suivre les progrès réalisés afin de réaliser nos ambitions
- 1 • Vous résolvez des problèmes complexes d'algorithme pour améliorer les performances du traitement et du stockage des données.
- Capacité à développer des solutions robustes et évolutives, en mettant l'accent sur l'optimisation des performances et la gestion de la scalabilité.
- 2 • Vous travaillez sur la conception et l'optimisation des structures de données, en particulier pour des systèmes distribués et à grande échelle.
 - Vous identifiez les opportunités d'amélioration de nos produits, services, processus, systèmes et outils.

Les Mots-Clés à Retenir

Problèmes complexes

Vos Responsabilités

- Vous travaillez directement avec nos responsables de catégories et vous les aidez à évaluer les segments de marché les plus prometteurs, à définir notre portefeuille et notre offre, à structurer le projet à lancer et à suivre les progrès réalisés afin de réaliser nos ambitions
- 1. Vous résolvez des problèmes complexes d'algorithmes pour améliorer les performances du traitement et du stockage des données.
Capacité à développer des solutions robustes et évolutives, en mettant l'accent sur l'optimisation des performances et la gestion de la scalabilité.
- 2. Vous travaillez sur la conception et l'optimisation des structures de données, en particulier pour des systèmes distribués et à grande échelle.
• Vous identifiez les opportunités d'amélioration de nos produits, services, processus, systèmes, et outils
- 3. Vous résolvez des problèmes complexes d'algorithmes pour améliorer les performances du traitement et du stockage des données.

Prêt(e) à relever le défi ? Rejoignez-nous et devenez acteur de votre carrière en développant une palette unique de compétences grâce à nos dispositifs de formation continue et à l'étranger.

Performances

Optimisation

Structures de données

Full Stack Developer – Data Struct...

Meta · Paris, Île-de-France, France (Sur site)

Postuler Enregistrer

Full Stack Developer – Data Structures, Algorithms Responsibilities:

1. Strong understanding of algorithms (binary search, sorting, trees, etc.) and data structures (arrays, lists, trees).
2. Familiarity with SQL and NoSQL databases (e.g., PostgreSQL, MongoDB).
Ability to write clean, efficient, and scalable code.
3. Strong problem-solving skills and the ability to optimize code for performance.
Set KPIs and goals, design and evaluate experiments, monitor key product metrics
understand root causes of changes in metrics
4. Experience with DevOps tools (Docker, Kubernetes) is a plus.

Minimum Qualifications:

Algorithmes

Clean-efficient-scalable code

Problem solving



L'Exemple d'Odoo

Pourquoi Investir dans le Clean Code ? L'Exemple d'Odoo

Do it cheaper

make better use of the available resources

Things that are cheaper

- Better algorithms
- Better data structures
- Stop using arrays for everything

Tools you can use

- Hash tables (objects)
- More hash tables (Map, Set)
- Exotic data structures (don't)

Don't do it

doing nothing is faster than doing something really fast

- Remove “zombie code”
- Only do as much as you need to
- Do it on the server instead

?

?

?

Things that are cheaper

- Better algorithms
- Better data structures
- Stop using arrays for everything

Tools you can use

- Hash tables (objects)
- More hash tables (Map, Set)
- Exotic data structures (don't)

Ces changements améliorent-ils réellement la qualité du logiciel, ou ne sont-ils qu'une illusion coûteuse en temps et en argent ?

Ce Que Montre l'Expérience d'Odoo

From 4.0s
all the way
down to
2.8s on
average.

Time to
load SO
(reload/FS)
reduced by
86%.

A 42%
decrease in
time to
load all
your apps.



Client Satisfait



A 18%
decrease
in JS+XML
code that
is used.

A 30%
decrease
in CSS
code that
is used.

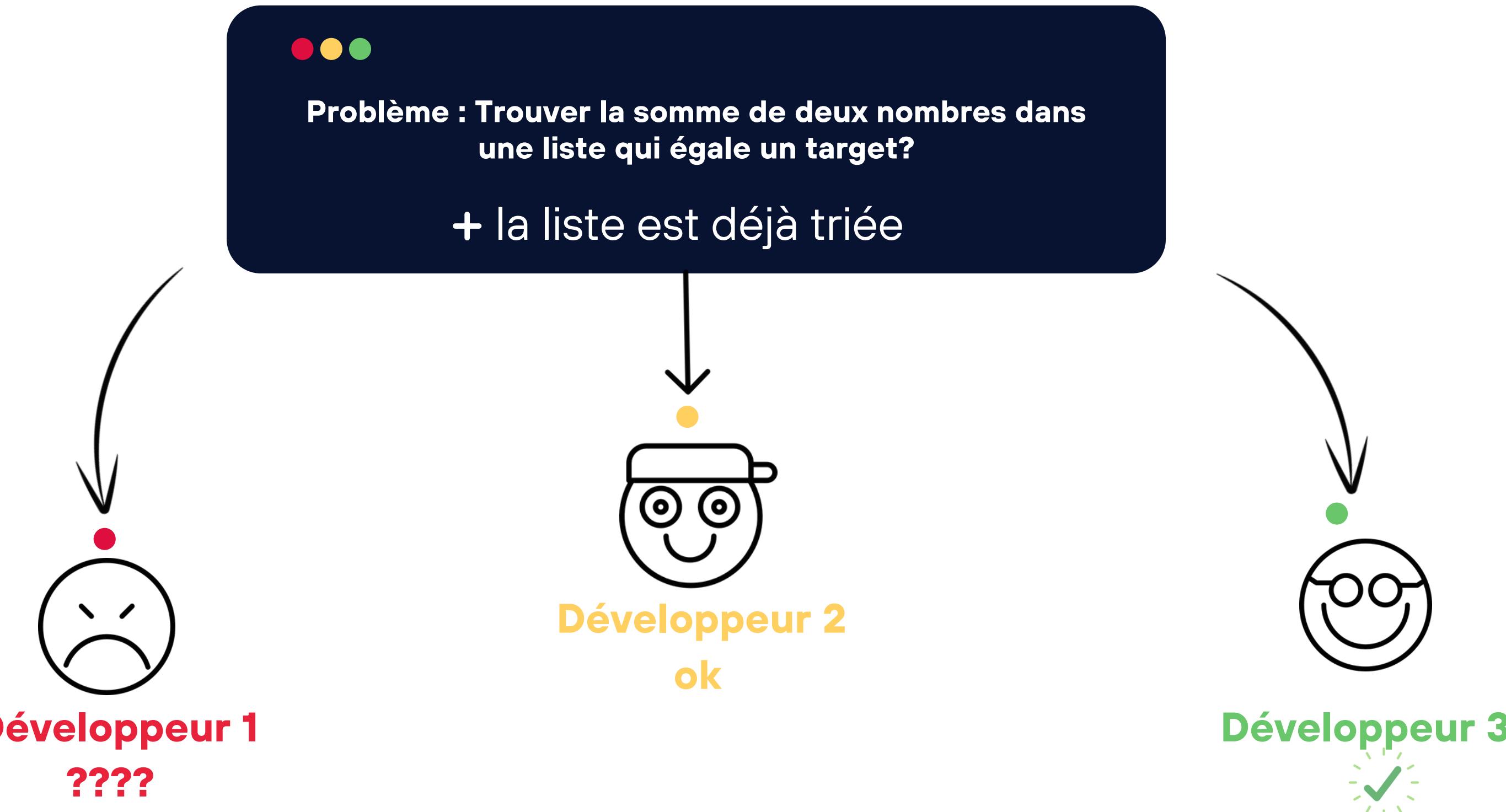
XHR RPC
Round Trip
7 --> 3
meaning a
223%
reduction.



The collage consists of four black and white photographs. Top-left: A person's hands typing on a laptop keyboard. Top-right: A woman in a hijab giving a presentation with a large screen showing two pie charts labeled 'Demographics'. Bottom-left: Five diverse people working together around a laptop. Bottom-right: Two people shaking hands over a desk with papers and a laptop.

Trois Développeurs, Trois Niveaux

Trois Développeurs, Trois Niveaux de Code



Développeur 1

????:Brute-force

Nom des variables ?

```
1 def f(l):  
2     for i in range(len(l)):  
3         for j in range(i + 1, len(l)):  
4             if l[i] + l[j] == 1256:  
5                 return l[i], l[j]  
6     return None  
7
```

Nom de la fonction ?

Commentaires ?

Complexité : ($O(n^2)$)

Target?

Temps d'exécution: 18.838437 secondes
Mémoire utilisée: 0.000000 Mo
Émissions de CO₂: 0.015699 kg

Développeur 2

ok : Recherche binaire

Obligation d'une liste triée

```
● ● ●  
1 def find_pair_with_binary_search(sorted_list, target_value):  
2     for index, current_value in enumerate(sorted_list):  
3         complement = target_value - current_value  
4  
5         complement_index = bisect.bisect_left(sorted_list, complement, index + 1)  
6  
7         if complement_index < len(sorted_list) and sorted_list[complement_index] == complement:  
8             return current_value, sorted_list[complement_index]  
9  
10    return None
```

Complexité : ($O(n \log(n))$)

Commentaires ?

Principe de la recherche binaire

Temps d'exécution: 7.021111 secondes
Mémoire utilisée: 0.000000 Mo
Émissions de CO₂: 0.005851 kg

Développeur 3

Table de hachage

Bien commenté



```
● ● ●  
1 """  
2 Trouve une paire de nombres dans la liste dont la somme est égale à target_sum.  
3  
4 Args:  
5     num_list (List[int]): Liste de nombres entiers.  
6     target_sum (int): Somme cible.  
7  
8 Returns:  
9     Optional[Tuple[int, int]]: Une paire (a, b) si elle existe, sinon None.  
10    """  
11  
12 def find_pair(numbers, target_sum):  
13     seen_numbers = {}  
14     for number in numbers:  
15         complement = target_sum - number  
16         if complement in seen_numbers:  
17             return complement, number  
18         seen_numbers[number] = True  
19     return None
```

Complexité : $(O(n))$

Mémoire?

Absence de type hinting

Temps d'exécution: 0.005946 secondes
Mémoire utilisée: 0.000031 Mo
Émissions de CO₂: 0.000005 kg

Clean code

Catégorie	Bonnes Pratiques	Exemple
Nommage	Utiliser des noms clairs et explicites.	<ul style="list-style-type: none">calculateTotal() au lieu de calc().userData au lieu de usrDta.getUser(), updateProfile().
Fonctions	Fonctions courtes et responsabilités uniques.	<ul style="list-style-type: none">calculateTotal(items) au lieu de tout faire dans processOrder(order).
DRY (Don't Repeat Yourself)	Éviter la duplication de code.	<ul style="list-style-type: none">Extraire le code répétitif dans une fonction réutilisable.
Commentaires	Expliquer le "pourquoi", pas le "comment". et Éviter les commentaires superflus.	<ul style="list-style-type: none">Pas besoin de i++ // increment i.
Conventions de Style	Suivre les conventions du langage.	<ul style="list-style-type: none">camelCase en JavaScript, snake_case en Python.

Green code

Optimisation temporelle

Bonnes Pratiques

Exemple

Réduction de la complexité algorithmique

Remplacer un algorithme $O(n^2)$ par un $O(n \log n)$
(tri rapide au lieu du tri par insertion)

Utilisation de structures de données adaptées

Utiliser un **HashMap** au lieu d'une liste pour des recherches rapides

Programmation concurrente et parallèle

Utiliser **async/await**, **ThreadPool**, ou **GPU** pour accélérer les calculs

Utiliser des fonctions rapides

map() au lieu de boucles classiques

Exécuter plusieurs tâches en même temps

async/await en **JavaScript** ou **multiprocessing** en **Python**, les **threads** en **Java**

Mémoriser les résultats réutilisables

cache en **Python** (`functools.lru_cache`)

Green code

Optimisation mémoire

Bonnes Pratiques

Exemple

Libérer la mémoire après usage

`del` en Python, `free()` en C

Stocker les données efficacement

Struct of Arrays (SoA) au lieu de Array of Structs (AoS) pour optimiser le cache

Charger uniquement ce qui est nécessaire

Lazy loading pour charger les images ou données à la demande

Éviter de stocker des données inutiles

Lire un fichier ligne par ligne au lieu de tout charger en RAM



Comparaison des Langages

Comparaison des Langages : Temps, Mémoire et Consommation d'Énergie

27 Langages de Programmation

Arbres Binaires

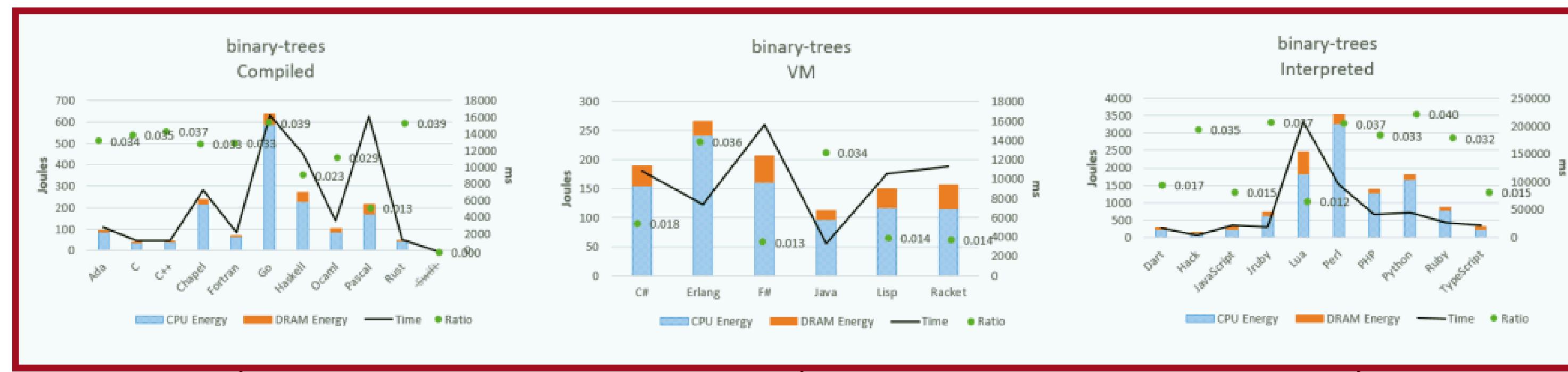
plus adapté à ton objectif ?

Time & Memory	Energy & Time	Energy & Memory	Energy & Time & Memory
C • Pascal • Go	C	C • Pascal	C • Pascal • Go
Rust • C++ • Fortran	Rust	Rust • C++ • Fortran • Go	Rust • C++ • Fortran
Ada	C++	Ada	Ada
Java • Chapel • Lisp • Ocaml	Ada	Java • Chapel • Lisp	Java • Chapel • Lisp • Ocaml
Haskell • C#	Java	Ocaml • Swift • Haskell	Swift • Haskell • C#
Swift • PHP	Pascal • Chapel	C# • PHP	Dart • F# • Racket • Hack • PHP
F# • Racket • Hack • Python	Lisp • Ocaml • Go	Dart • F# • Racket • Hack • Python	JavaScript • Ruby • Python
JavaScript • Ruby	Fortran • Haskell • C#	JavaScript • Ruby	TypeScript • Erlang
Dart • TypeScript • Erlang	Swift	TypeScript	Lua • JRuby • Perl
JRuby • Perl	Dart • F#	Erlang • Lua • Perl	
Lua	JavaScript	JRuby	
	Racket		
	TypeScript • Hack		
	PHP		
	Erlang		
	Lua • JRuby		
	Ruby		

Énergie vs Temps vs Mémoire

Énergie (J) = Puissance (W) × Temps (s)

Binary Trees : Quel Langage est le Plus Efficace ?

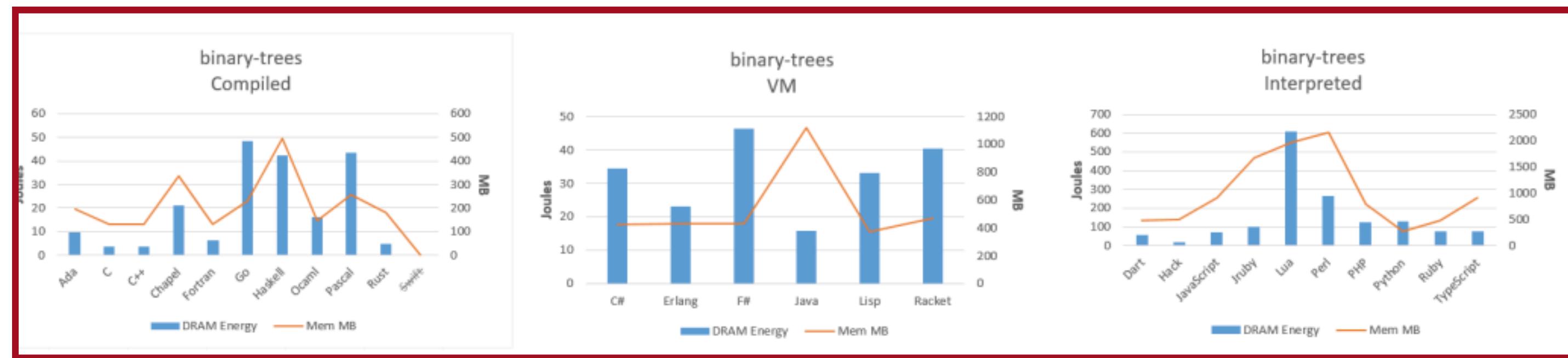


- ✓ Convertis en code machine avant exécution grâce à un compilateur.
- ✓ Exécution très rapide car directement compris par l'ordinateur.
- ✓ Besoin de recompiler si on change de machine (Windows, Linux, etc.).

- ✓ Compilés en un code intermédiaire (bytecode).
- ✓ Exécutés sur une machine virtuelle (JVM pour Java, .NET pour C#).
- ✓ Portables, car le même code fonctionne sur différentes plateformes.

- ✓ Lus ligne par ligne par un interpréteur au moment de l'exécution.
- ✓ Plus lents que les compilés car pas convertis en avance.
- ✓ Faciles à modifier et tester, car pas besoin de compilation.

Binary Trees : Quel Langage est le Plus Efficace ?

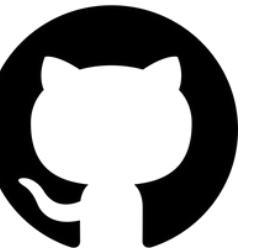


- ▶ L'énergie consommée dépend principalement du temps d'exécution, qui suit une relation linéaire avec la consommation énergétique.
- ▶ La mémoire utilisée n'a pas d'impact direct et systématique sur la consommation d'énergie, même si elle peut influencer indirectement les performances et donc le temps d'exécution.



Ma Méthodologie de Veille Technologique

ApiGithub



```
1 import requests
2 import json
3
4 query = "clean code"
5 url = f"https://api.github.com/search/repositories?q={query}&sort=
6
7 headers = {
8     "Accept": "application/vnd.github.v3+json",
9 }
10
11 def search_github_repositories():
12     response = requests.get(url, headers=headers)
13
14     if response.status_code == 200:
15         data = response.json()
16         repositories = data.get("items", [])
17
18         if not repositories:
19             print("Aucun repository trouvé.")
20             return
21
22         repo_list = []
23         for repo in repositories:
24             repo_data = {
25                 "name": repo.get("name"),
26                 "url": repo.get("html_url"),
27                 "stars": repo.get("stargazers_count"),
28                 "forks": repo.get("forks_count"),
29                 "topics": repo.get("topics", []),
30                 "description": repo.get("description"),
31                 "language": repo.get("language"),
32                 "owner": repo.get("owner", {}).get("login"),
33                 "created_at": repo.get("created_at"),
34                 "updated_at": repo.get("updated_at"),
35                 "license": repo.get("license", {}).get("name") if
36             }
37             repo_list.append(repo_data)
38
39         with open("github_clean_code_repos.json", "w", encoding="u
40             json.dump(repo_list, json_file, indent=4, ensure_ascii
41
42             print("Les résultats ont été enregistrés dans 'github_clea
43
44 else:
45     print(f"Erreur {response.status_code}: Impossible de récup
46
47 if __name__ == "__main__":
48     search_github_repositories()
```

```
1 {
2     "name": "clean-code-javascript",
3     "url": "https://github.com/ryanmcdermott/clean-code-javascript",
4     "stars": 92347,
5     "forks": 12370,
6     "topics": [
7         "best-practices",
8         "clean-architecture",
9         "clean-code",
10        "composition",
11        "inheritance",
12        "javascript",
13        "principles"
14    ],
15    "description": "Clean Code concepts adapted for JavaScript",
16    "language": "JavaScript",
17    "owner": "ryanmcdermott",
18    "created_at": "2016-11-25T22:25:41Z",
19    "updated_at": "2025-02-04T10:05:20Z",
20    "license": "MIT License"
21 },
```

les dépôts les plus populaires sur le clean et green code

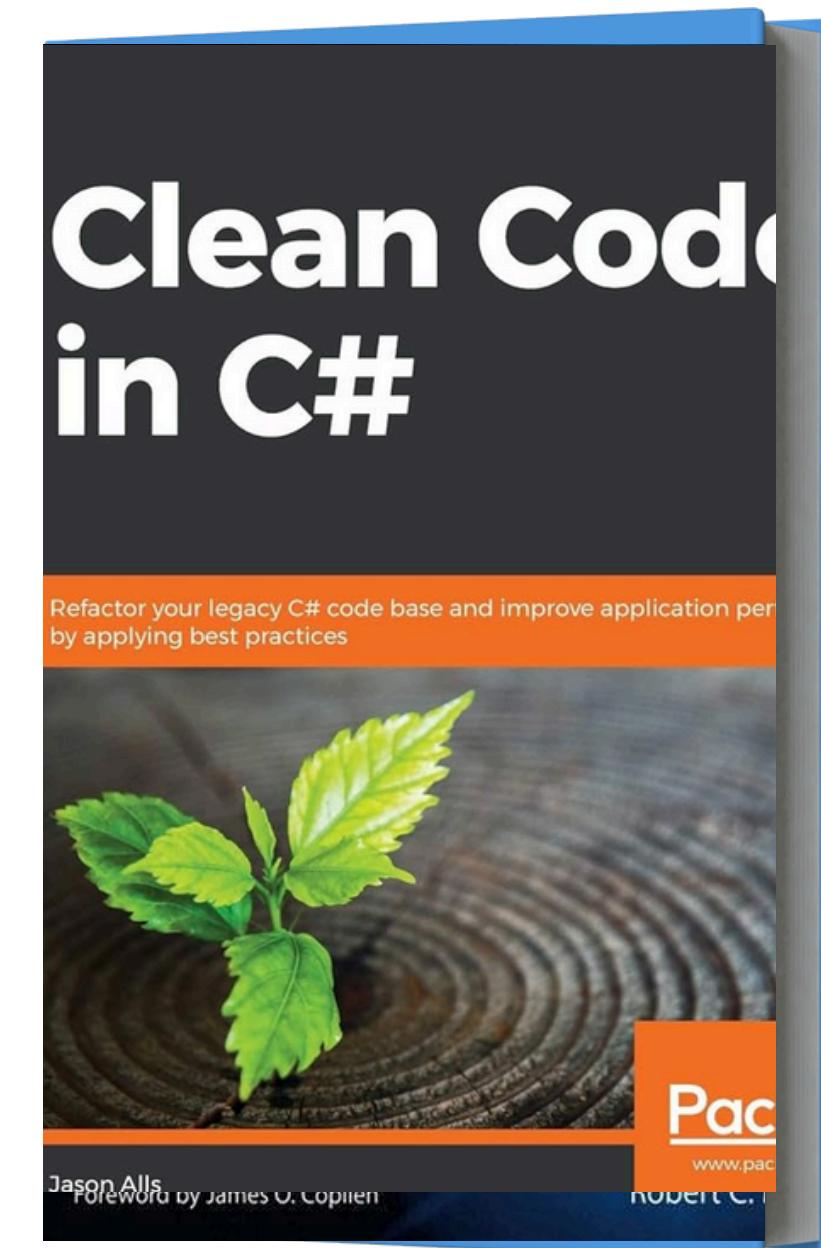
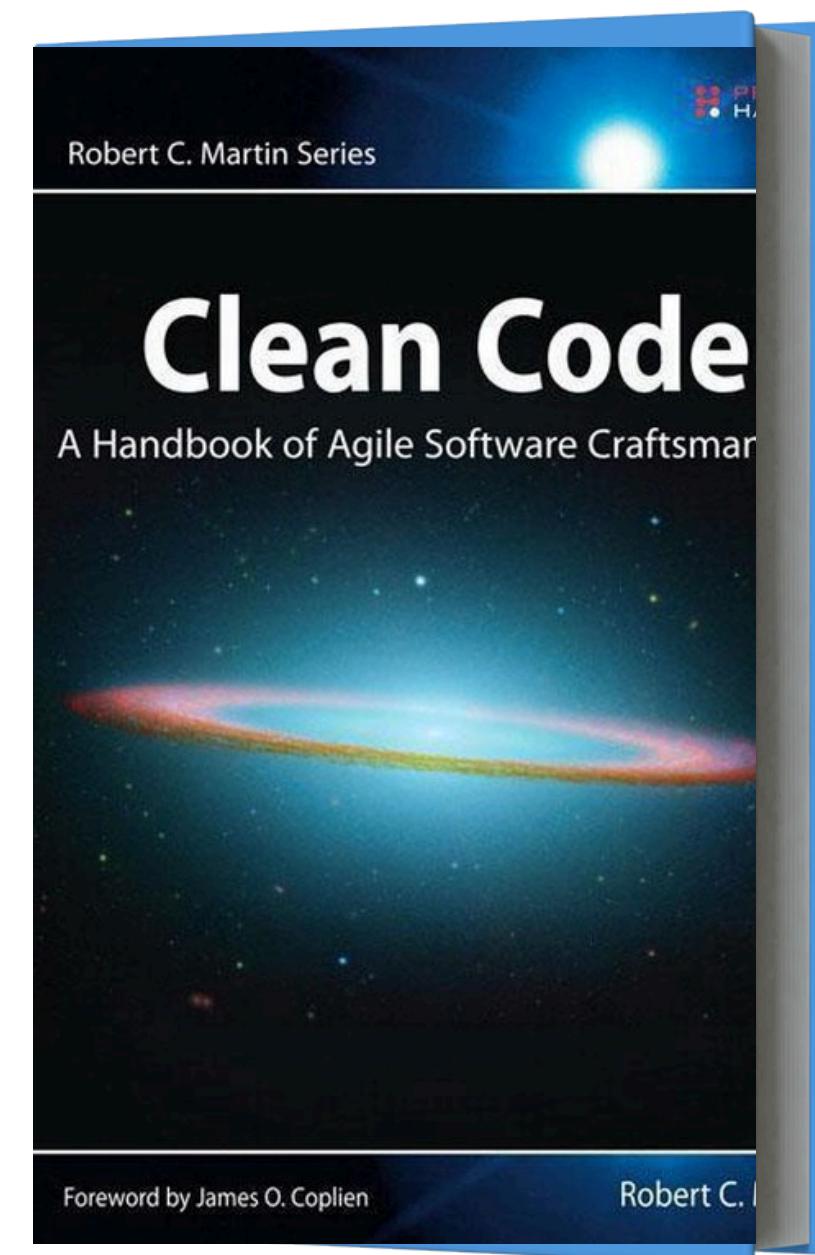
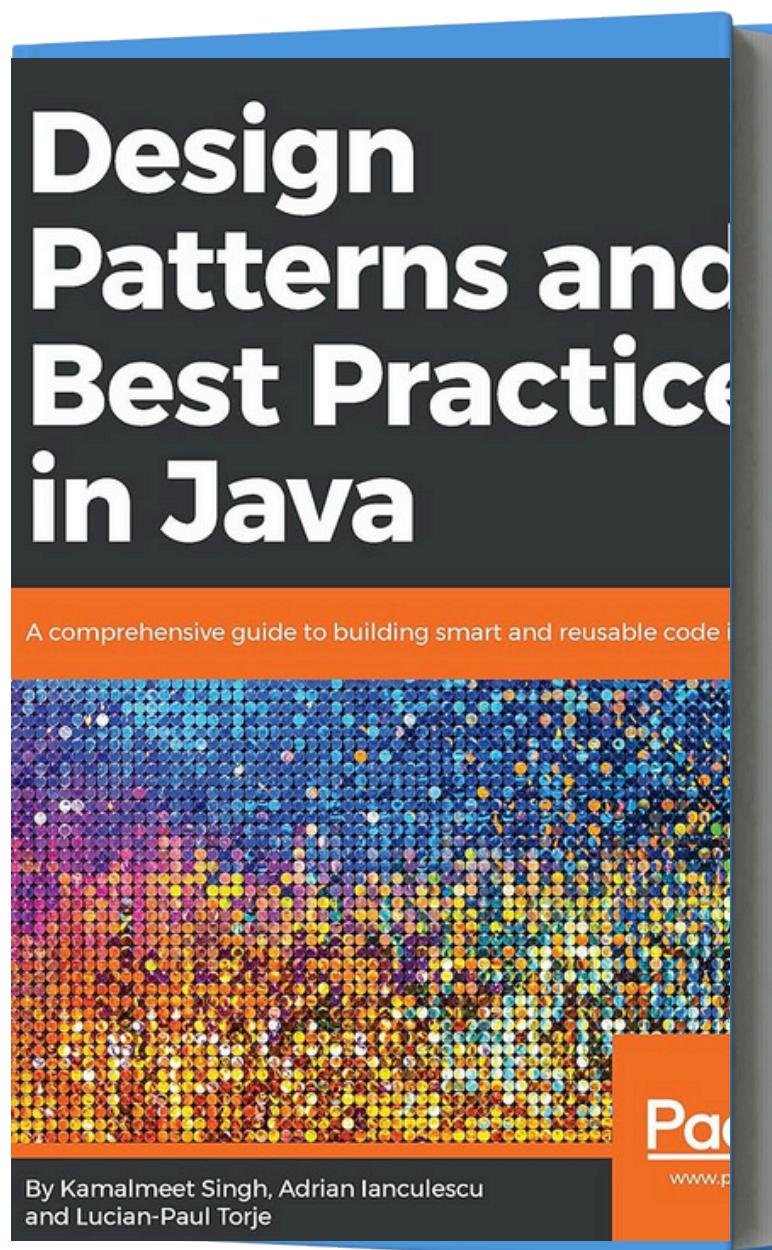




Google Scholar

- ▶ **Clean Code AND "Green Code"**
- ▶ **"Software Optimization" AND "Energy Efficiency"**
- ▶ **"Clean Code" AND "Best Practices" AND "Sustainability"**
- ▶ **intitle:"Clean Code"**

Des livres



Mes Sources



```
1 #Si vous voulez plus d'informations sur le sujet, voici quelques liens utiles:  
2 https://sites.google.com/view/energy-efficiency-languages  
3 https://www.green-coding.io/case-studies/energy-efficiency-python/  
4 https://www.linkedin.com/feed/update/urn:li:activity:7196781299804631040  
5 https://www.youtube.com/watch?v=dDwr-\_b2geU&t=309s  
6 https://www.slideshare.net/slideshow/clean-code-256238273/256238273  
7 https://github.com/jnguyen095/clean-code/blob/master/Clean.Code.A.Handbook.of.Agile.Software.Craftsmanship.pdf  
8 https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=http://repo.aassfxxx.infos.st/docs/Coder%2520Proprement.  
9
```

Original work in SLE'17

The tools and graphical data pointed by this page are included in the research paper "Energy Efficiency across Programming Languages: How does Energy, Time and Memory Relate?", accepted at the International...

google.com

Clean code-Green code

Merci pour votre attention !

A blurred screenshot of a computer screen showing a terminal window with code and a code editor window with Ruby code.

```
require 'copybara/rspec'  
require 'copybara/rails'  
  
Copybara.javascript_driver = :webkit  
Category.delete_all; Category.create!({  
  name: "Category 1",  
  description: "Category 1 description"  
})  
Shoulda::Matchers.configure do |config|  
  config.integrate do |with|  
    with.test_framework :rspec  
    with.library :rails  
  end  
end  
  
# Add additional require statements below this line if needed  
  
# Requires supporting files within the same directory as this file if  
# specified.  
# spec/support/ and its subdirectories will be added to the search path.  
# run as spec files by default.  
# in _spec.rb will both be required.  
# run twice. It is recommended that you do not name files  
# with _spec.rb. You can  
# specify this one manually per file in the RSpec manifest:  
# RSpec.configure do |config|  
#   config.manifest = false  
# end
```