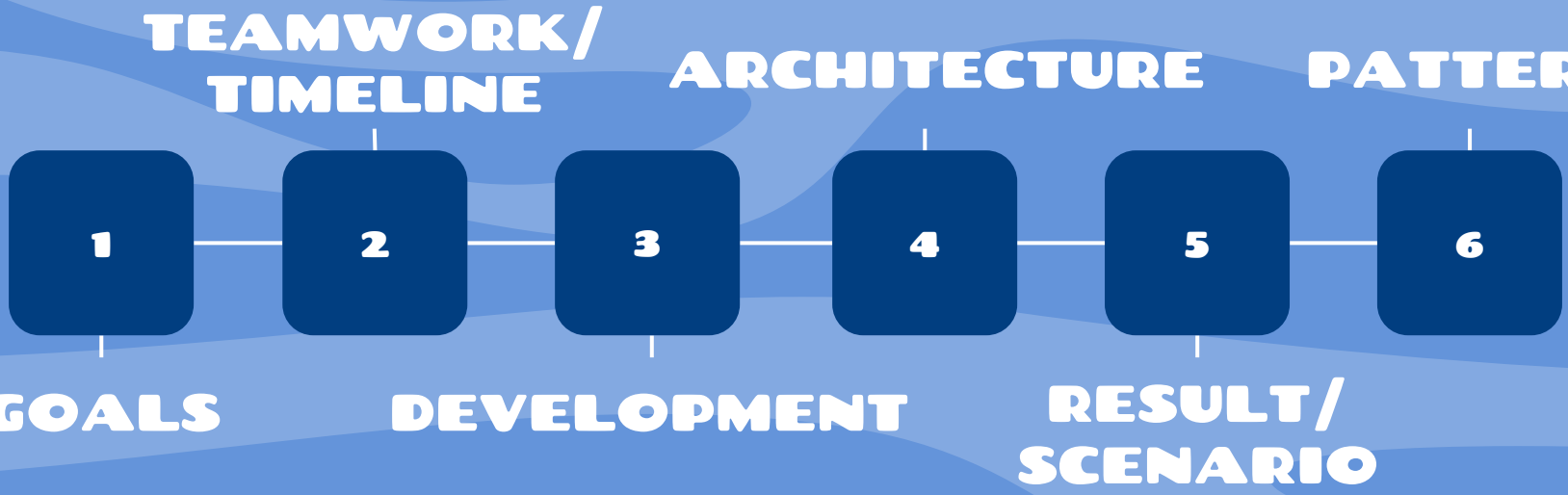
탄소를 JAVA라!

Team 1.

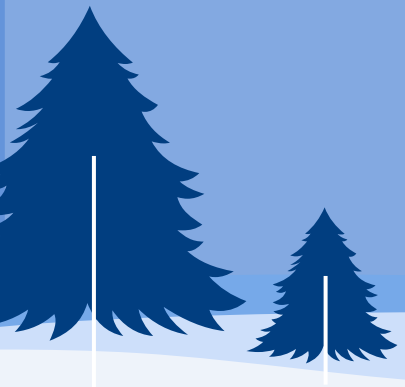
김민지 | 김찬용 | 안상현 | 양승빈 | 윤시형 | 임동준 | 최경식

CONTENTS


1
GOALS
2
TEAMWORK/
TIMELINE
3
DEVELOPMENT
4
ARCHITECTURE
5
RESULT/
SCENARIO
6
PATTERN

01

GOALS



GOALS

1.

탄소 배출량
웹 사이트 제작

2.

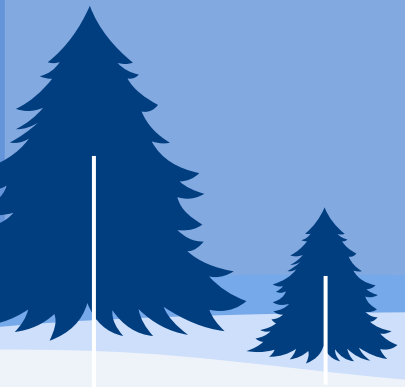
탄소 배출량 시각화

3.

그린화 패턴 탐색

02

TEAMWORK / TIMELINE



TEAMWORK

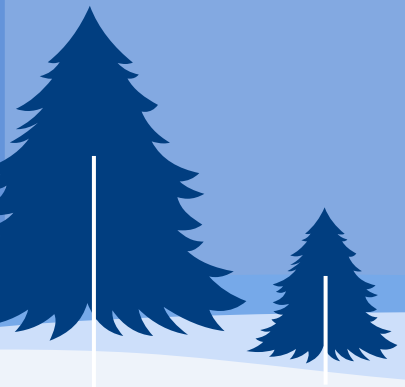
GROUP	PEOPLE
Front-end	김민지, 임동준, 최경식
Back-end	김찬용, 안상현
Document & Pattern	윤시형, 양승빈

TIMELINE

4	5	6	7	8	9	10	11	12	13	14	15
요구사항 정의											
	디자인 정의										
		프론트엔드 개발									
			백엔드 개발								
					통합						
						자바 패턴 탐색					
									테스트		발표

03

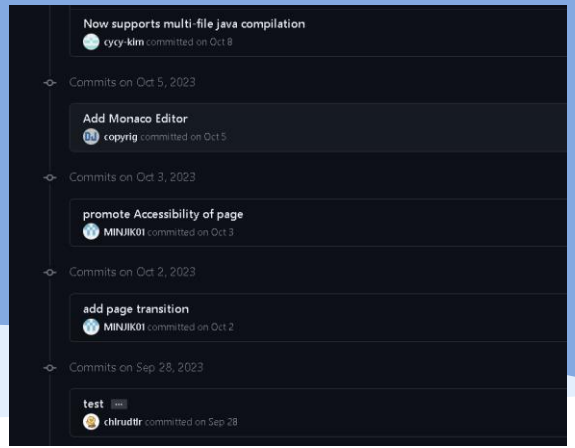
DEVELOPMENT



DEVELOPMENT

Cooperation Tool

- 버전 컨트롤 및 전반적인 개발
- 개발 일정 및 회의록 관리



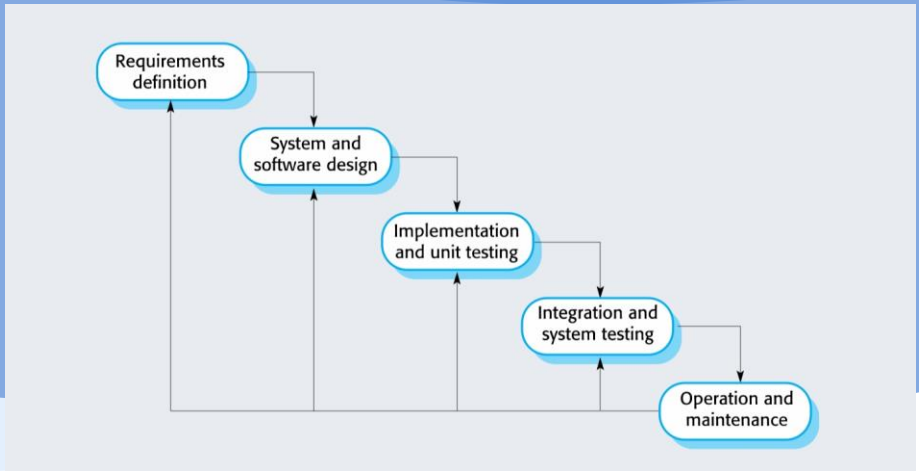
GitHub

- 패턴
- 후보군
- 제안서 피드백
- 기술 스펙 예시
- 브레인스토밍 예시
- PRD 예시
- 문서 템플릿으로 시작하기

Notion

DEVELOPMENT

- 고객의 요구사항이 명확
- 제안서 발표 피드백을 반영하여 Plan driven development process 채택
- 요구 사항 구현에 초점을 맞춰 개발 진행



DEVELOPMENT

User

Web Client

Java code input & result page

Display result with server info

Send code

API Server

System info & Runtime connection

Result of compile and run

Run code

Java system

JDK runtime

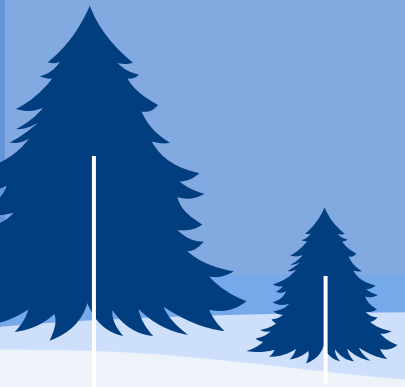
React

FastAPI

Java

04

ARCHITECTURE



ARCHITECTURE - Frontend

제목

코드 입력

탄소배출량, 성공여부, 실행시간

탄소배출량을 일상 속 요소로 환산한 값
- 승용차가 몇 m 이동한 것인지
- 휴대전화를 몇 % 충전한 것인지
- 에어컨을 몇 초 켠 것인지
- 나무가 몇 초 흡수할 양인지

서버 사양
- CPU
- 메모리
- 서버 소재지

Java standard output

Frontend

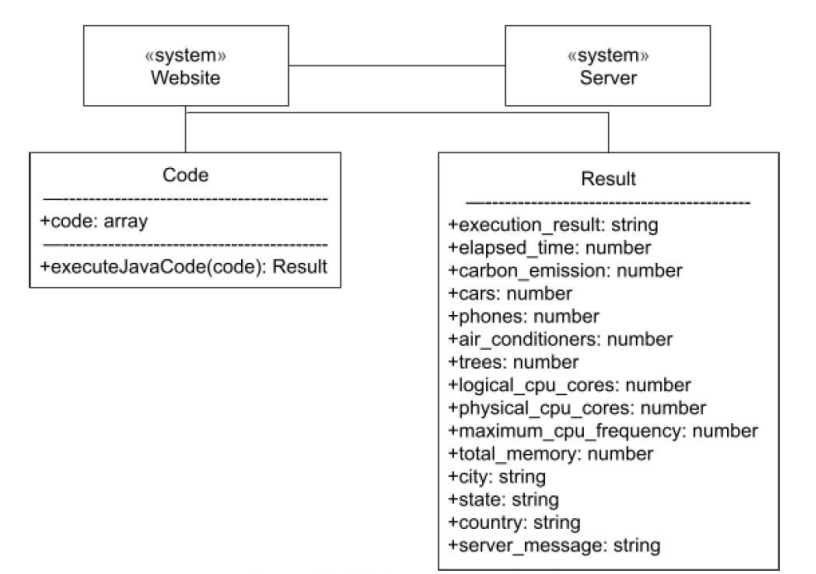


Image 4.1: Class Diagram - Front end

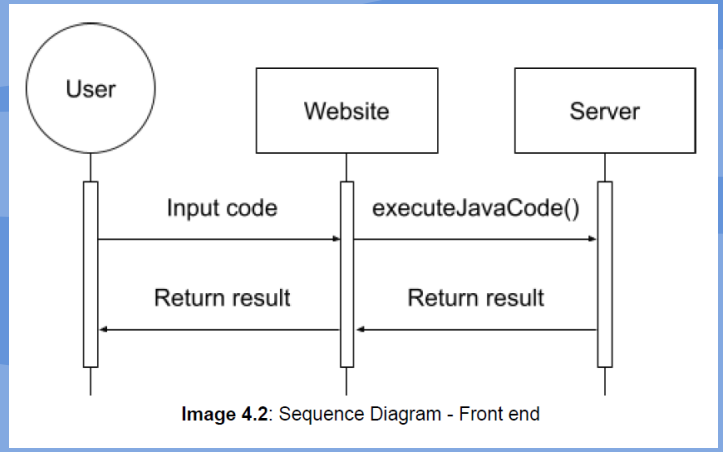
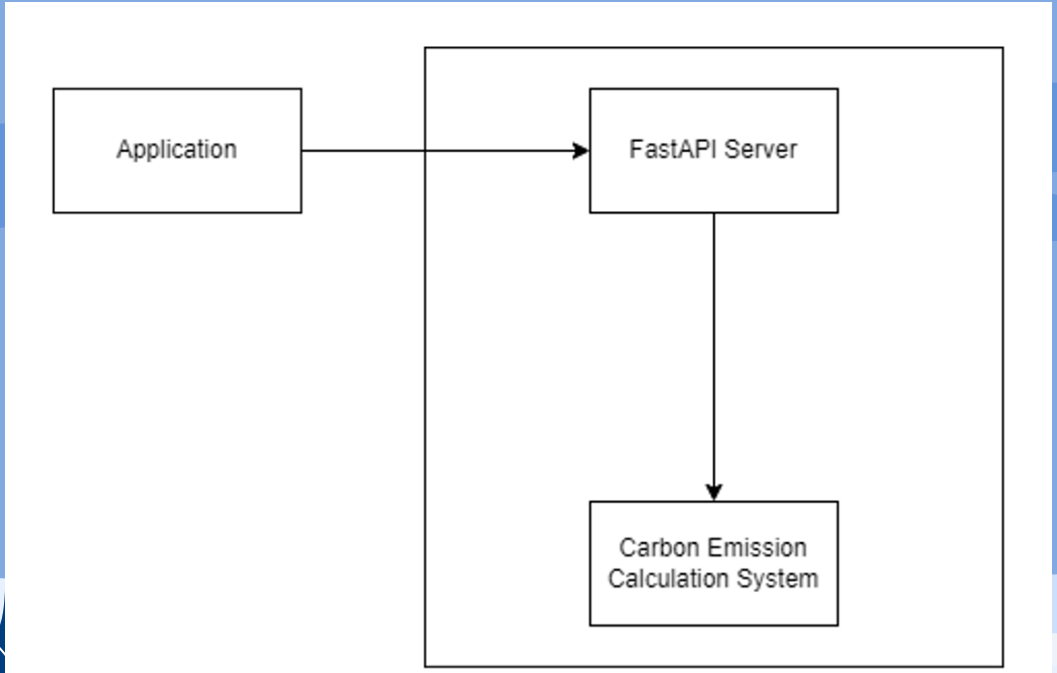


Image 4.2: Sequence Diagram - Front end

Result:

- 실행 결과 및 소요시간
- 탄소배출량
  - 생활 속 기준으로 환산한 값
- 서버 사양 및 소재지 정보
- Java의 standard output

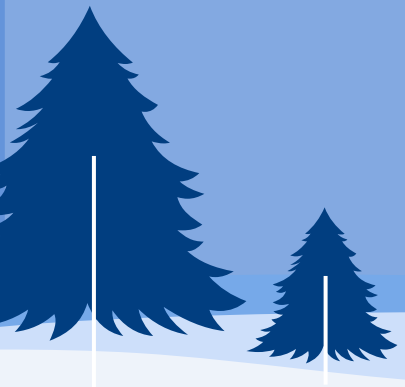
ARCHITECTURE - Backend



1. FastAPI를 통한 서버 호출
2. 탄소배출량 계산 시스템 실행

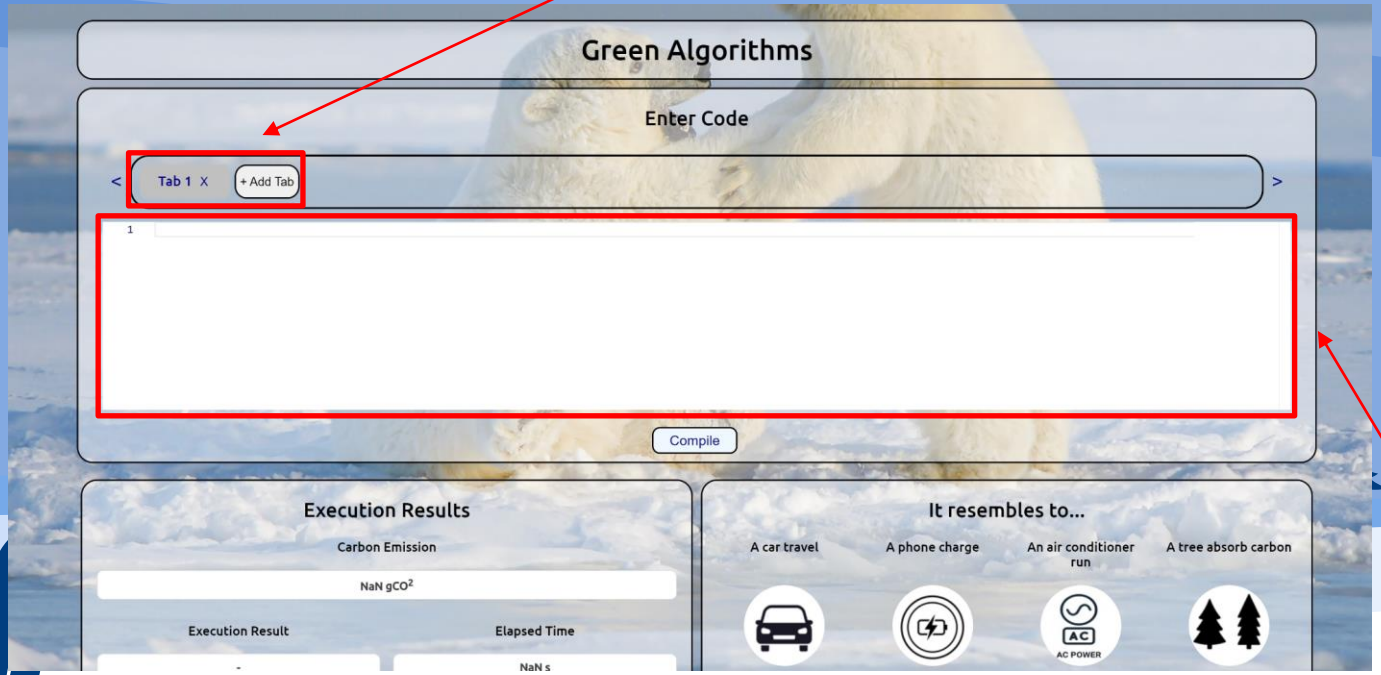
05

RESULT / SCENARIO



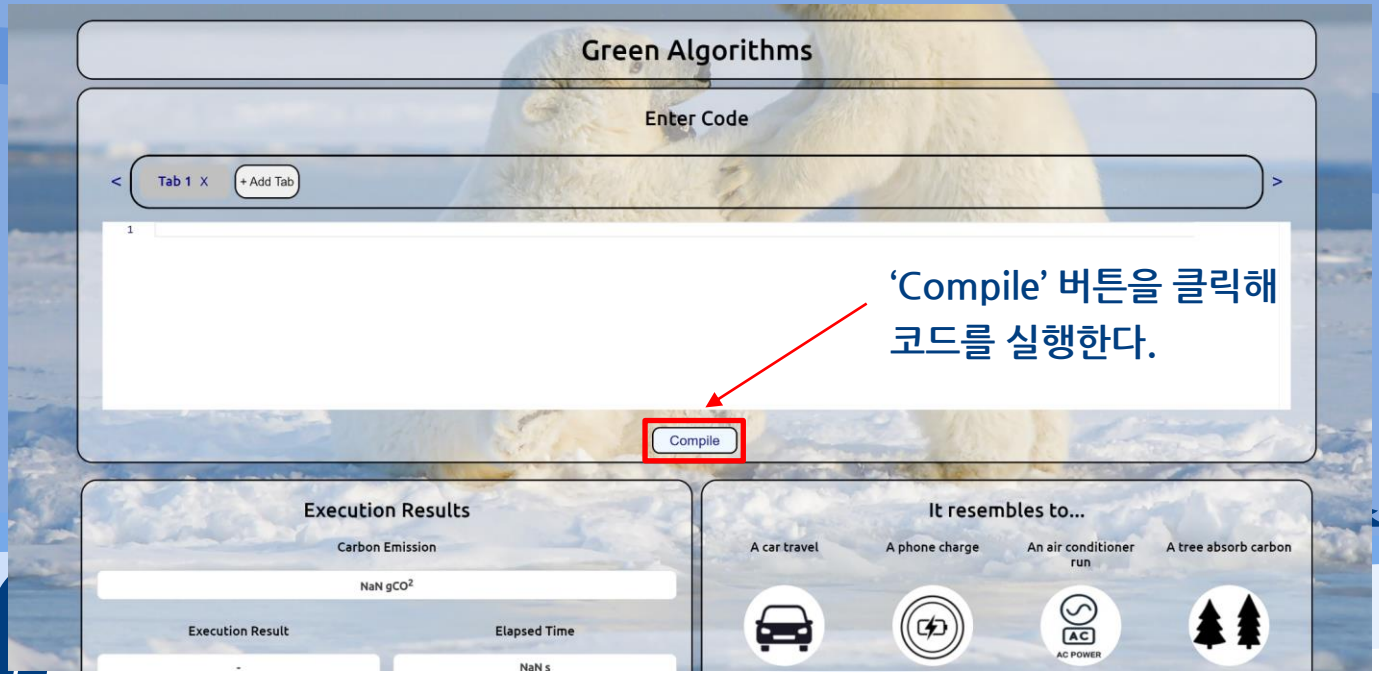
RESULT / SCENARIO

Add Tab 버튼을 클릭해서 새로운 탭을
추가하거나 x버튼을 눌러 탭을 지울 수 있다.



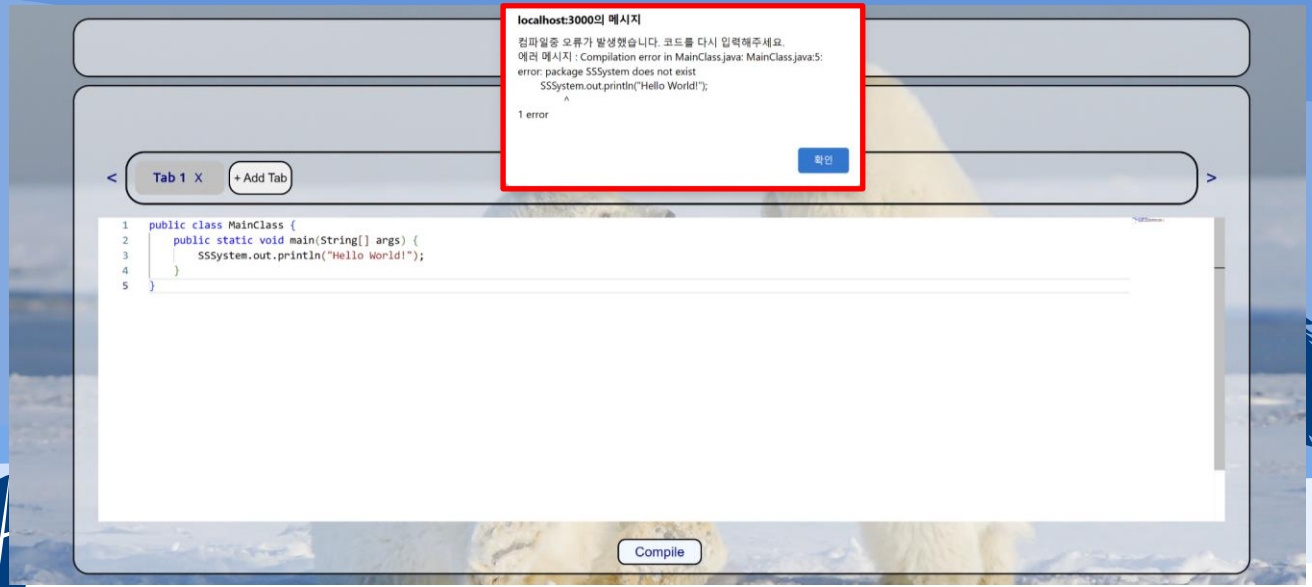
코드 입력

RESULT / SCENARIO



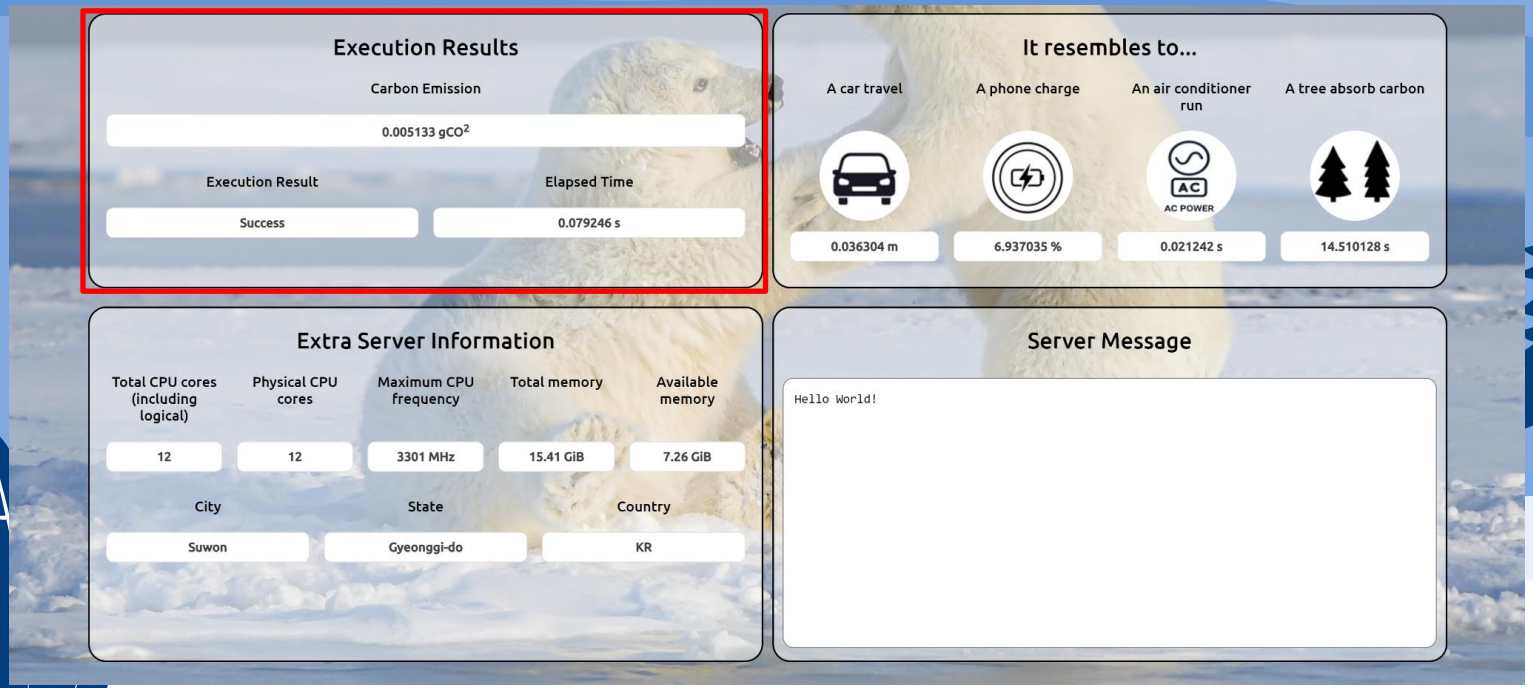
RESULT / SCENARIO

실행 과정에서 오류가 났을 때의 메시지



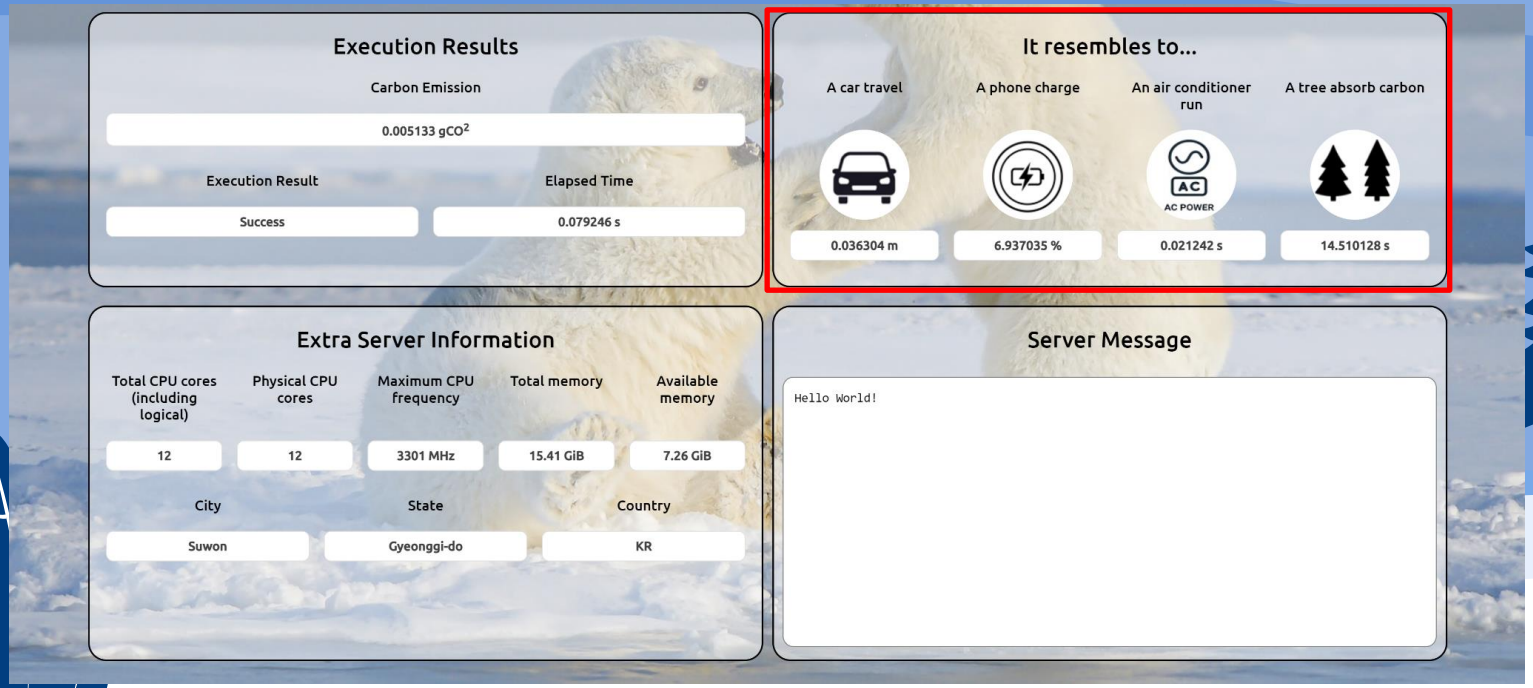
RESULT / SCENARIO

실행 결과



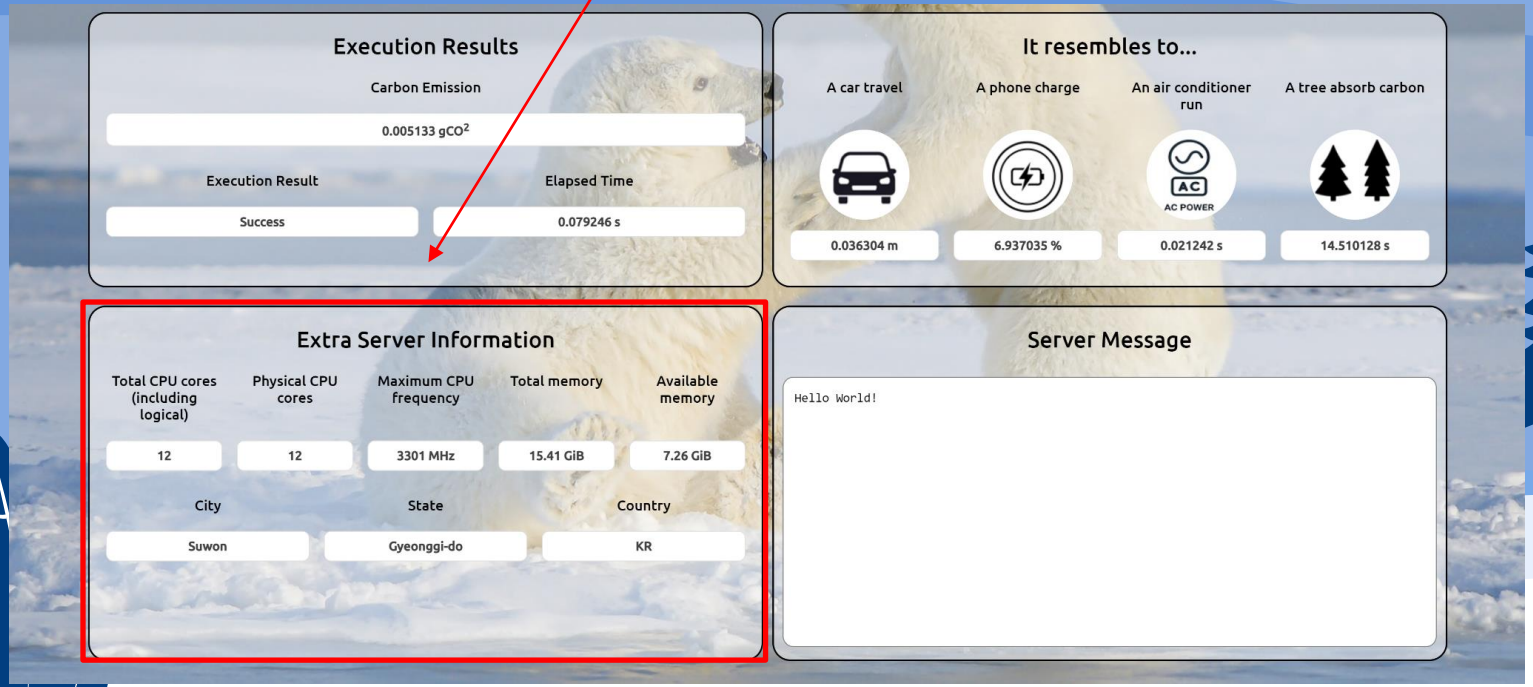
RESULT / SCENARIO

탄소 배출량을 다른 값으로 환산



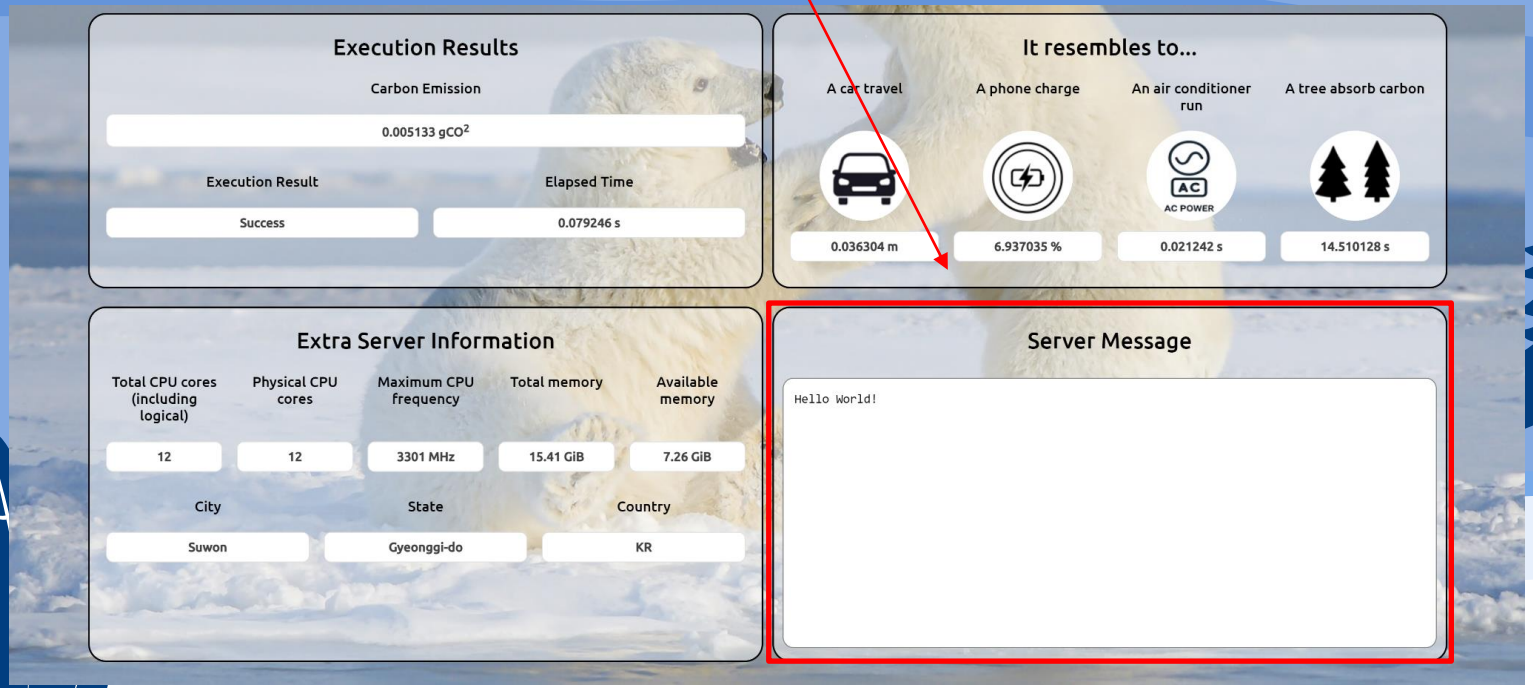
RESULT / SCENARIO

서버 환경



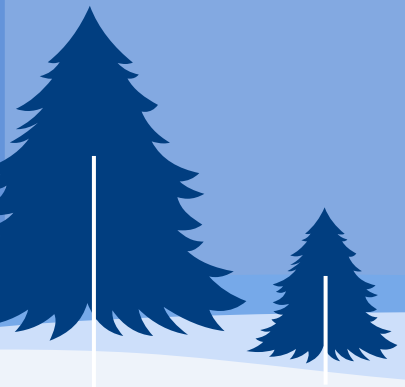
RESULT / SCENARIO

표준 출력 결과



06

PATTERN



PATTERN

Carbon Footprint

= Energy Needed × Carbon Intensity

Energy Needed

= Runtime × (Power draw for cores × Usage × Power draw for memory) × PUE × PSF

PATTERN

Carbon Footprint

= Energy Needed × Carbon Intensity

Energy Needed

= Runtime × (Power draw for cores
× Usage × Power draw for memory)
× PUE × PSF

: Code Independent

: Code Dependent

PATTERN

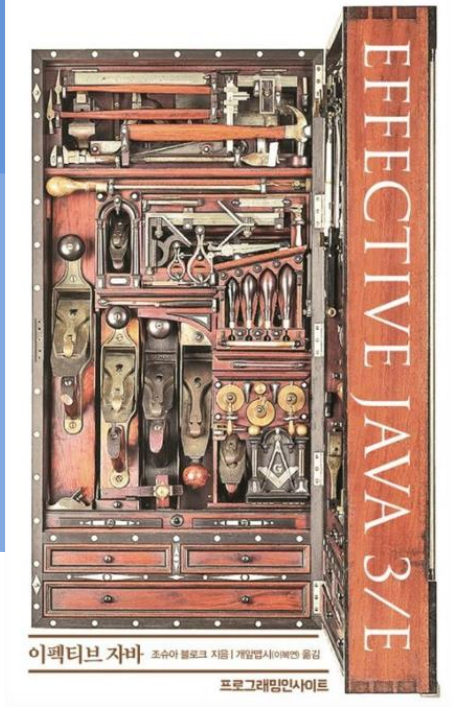
Java Code

Team 1

Any Language

Team 2

PATTERN



QUESCOL

B. Tech CSE Second Year
B. Tech CSE Third Year
B. Tech CSE Fourth Year
Interview Preparation

Data Structure Units

Efficiency of an Algorithm with the help of examples

jbloch / effective-java-3e-source-code

Code | Issues 11 | Pull requests 5 | Actions | Projects | Security

effective-java-3e-source-code Public

Watch 227

master | 1 branch | 0 tags | Go to file | Add file | Code

jbloch Fixed double check idiom exa... | 1 | bdc828a on Sep 12, 2019 | 10 commits

src/effectivejava	Fixed double check idiom example from page 3....	4 years ago
.gitignore	Initial commit of Effective Java, 3e source code,...	5 years ago
README.md	Fixed markdown.	5 years ago

README.md

Effective Java, Third Edition

MaheshB77 / freecodecamp-java-algorithms

Code | Pull requests | Actions | Projects | Security | Insights

freecodecamp-java-algorithms Public

forked from williamfiset/Algorithms

master | 1 branch | 0 tags | Go to file | Add file | Code

This branch is 168 commits behind williamfiset:master.

williamfiset Update README.md 6bd2d44 on Jun 12, 2021 1,463 commits

.github/workflows	Check workflows (williamfiset#160)	3 years ago
gradle/wrapper	Added Remove Method To Red Black Tree (will...	3 years ago
misc/images	Added comment img	3 years ago
references	Add aho Corasick reference pdf	3 years ago
slides	WMCM	2 years ago
src	WMCM	2 years ago

Products | Solutions | Pricing | Resources | About | Free Trial

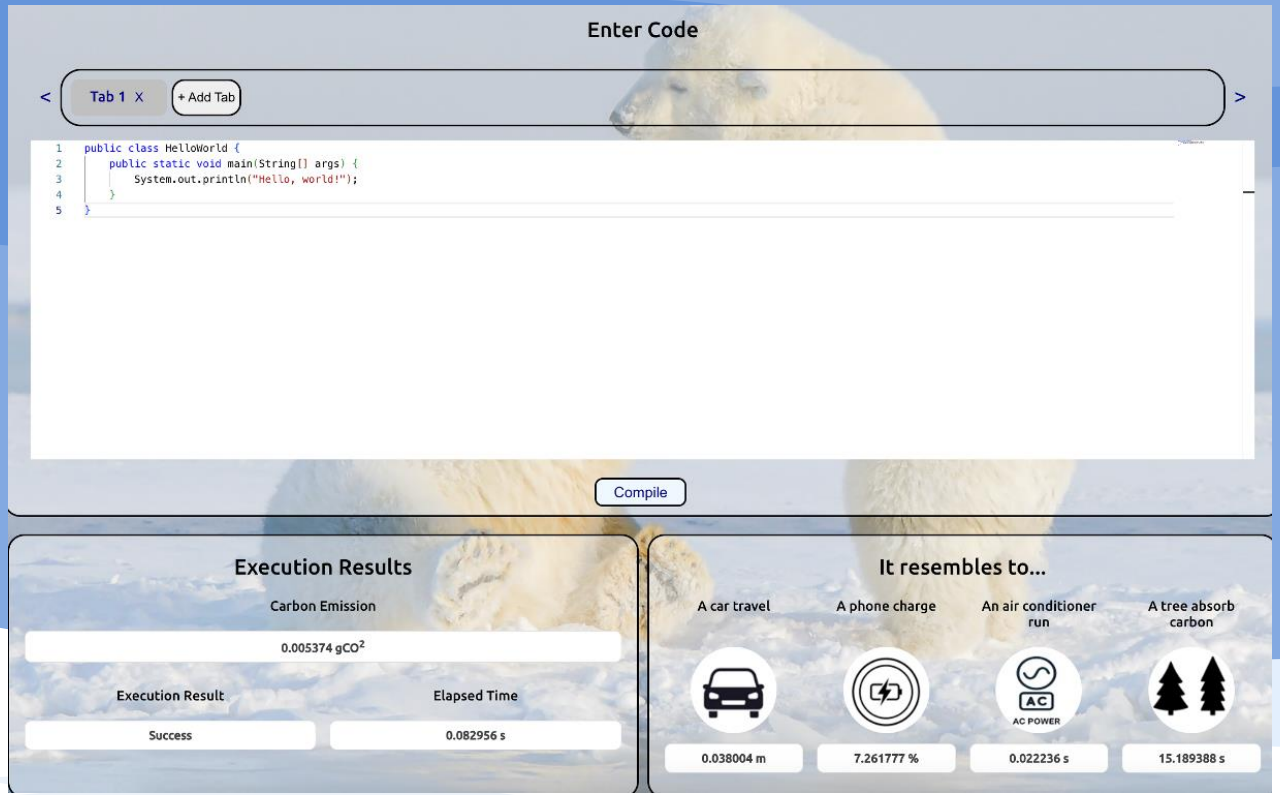
IN THIS BLOG POST

1. Making Java Applications Run Faster
2. 6 Tips for Application Developers to Make Java Applications Faster
3. Select the Java collection to use in your application carefully

Making Java Applications Run Faster

Application developers and application operations personnel are together responsible for e
that Java applications perform well. In an earlier blog, we had discussed 7 configuratio
Application Operations teams can use to make their Java applications high-performing. In th
we will focus on Application Developers and discuss 6 ways in which they can enhan
performance of their Java applications and make Java run faster.

PATTERN


Enter Code
Tab 1 X
+ Add Tab
public class HelloWorld {
public static void main(String[] args) {
System.out.println("Hello, world!");
}
}
Compile
Execution Results
Carbon Emission
0.005374 gCO2
Execution Result
Success
Elapsed Time
0.082956 s
It resembles to...
A car travel
A phone charge
An air conditioner run
A tree absorb carbon
AC
AC POWER
0.038004 m
7.261777 %
0.022236 s
15.189388 s

REFERENCE

- https://calculator.green-algorithms.org/
- https://dahye-jeong.gitbook.io/java/java/effective_java
- https://github.com/jbloch/effective-java-3e-source-code
- https://github.com/MaheshB77/freecodecamp-java-algorithms
- https://quescol.com/data-structure/efficiency-of-an-algorithm
- https://www.eginnovations.com/blog/6-tips-fast-java-applications/

THANKS!

Do you have any questions?