Java Programming II

Lab2

514770-1
Fall 2025
9/17/2025

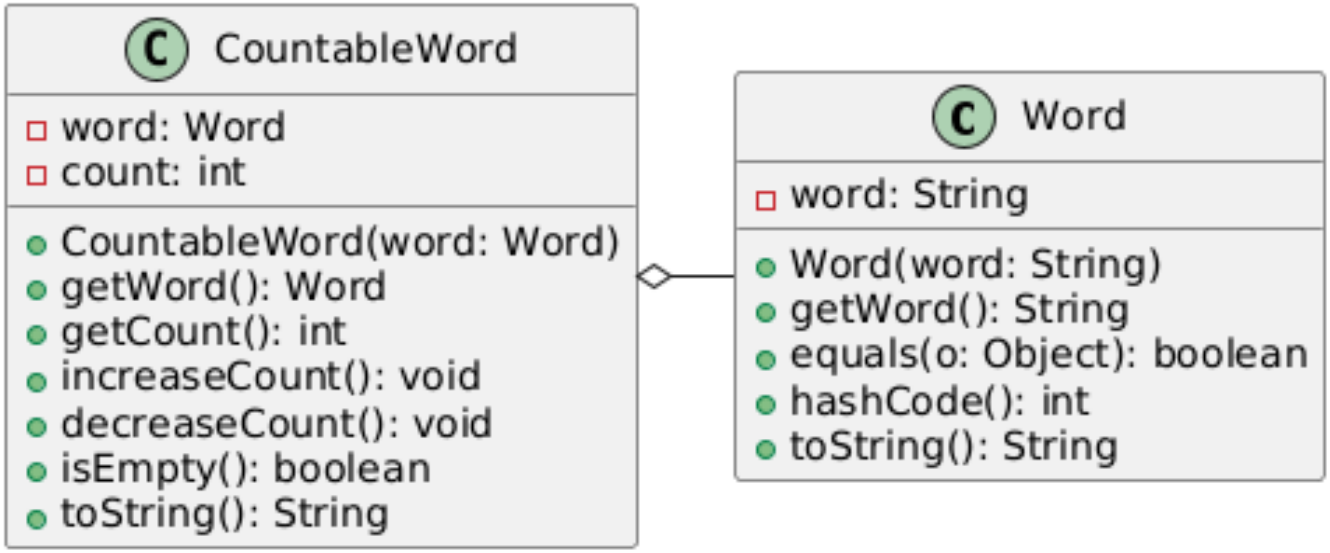
Kyoung Shin Park
Computer Engineering
Dankook University

Lab2

- Strategy 패턴을 사용하여 Preprocessor 클래스에서 다양한 알고리즘으로 text 처리하는 프로그램을 작성하라.
  - List<ITextNormalizer> textNormalizers 텍스트 정규화
    - NFKCTextNormalizer 문자열을 유니코드 정규화(NFKC)
    - LowercaseTrimTextNormalizer 공백제거(trim) 및 소문자화(lowercase)
    - AsciiFoldingNormalizer 악센트/다이어크리틱 제거(é→e, ö→o) 공백제거 소문자화
    - SimpleStemNormalizer 공백제거 및 소문자화 + 소유격('s|'s|s') 복수형(ies, s, es) 시제(ed)/동명사(ing) 제거
  - ITokenizer tokenizer
    - RegexTokenizer 정규식패턴을 사용해 토큰화
    - WhitespaceTokenizer 공백을 기준으로 문자열 토큰화

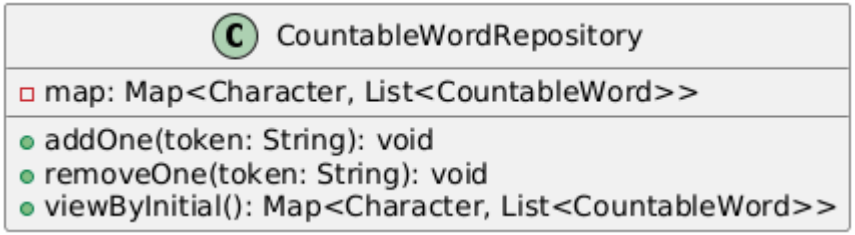
Lab2

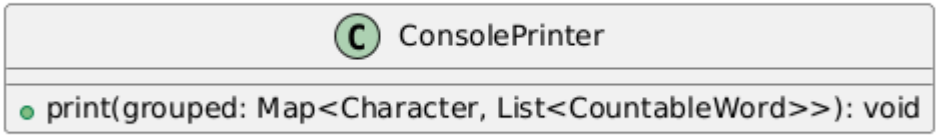
- CountableWord & Word 클래스



Lab2

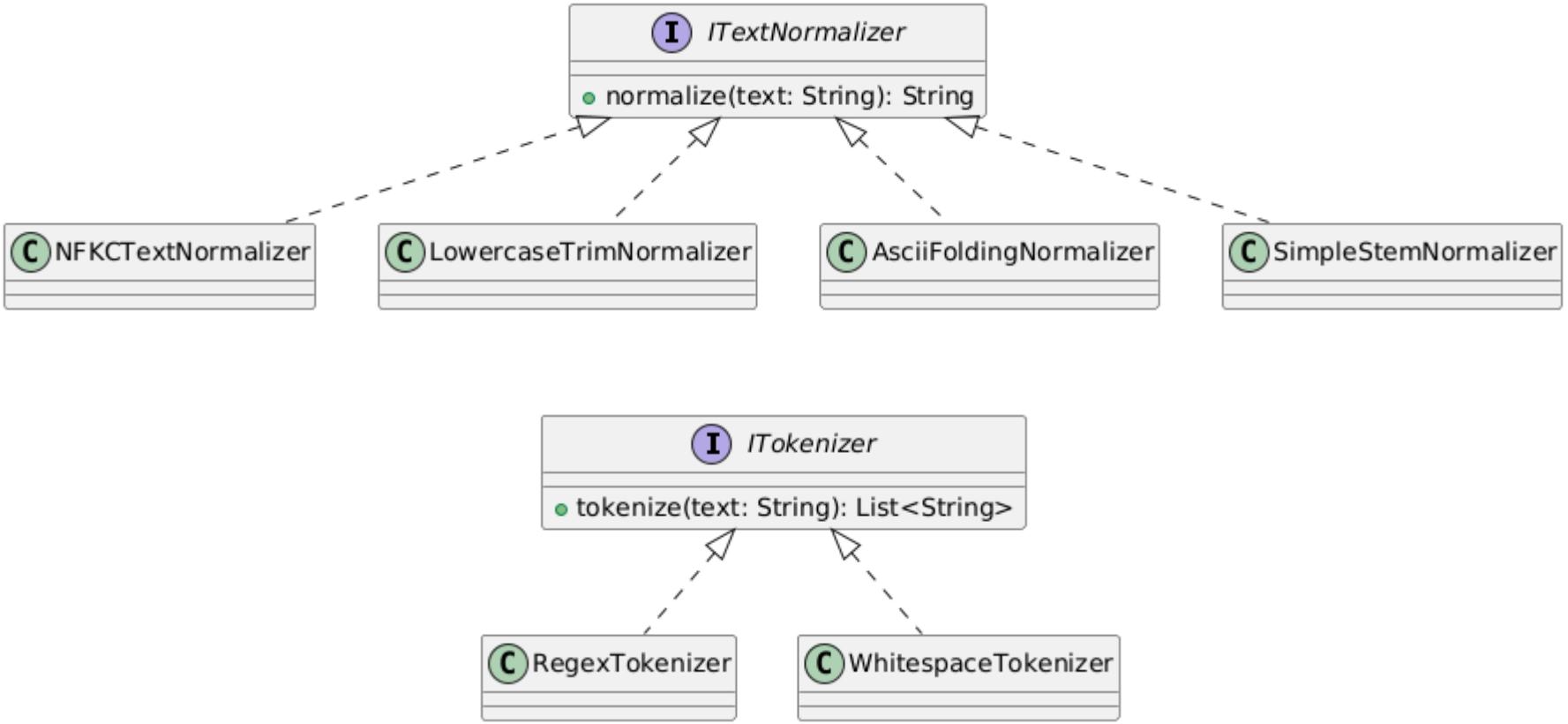
- CountableWordRepository & ConsolePrinter 클래스





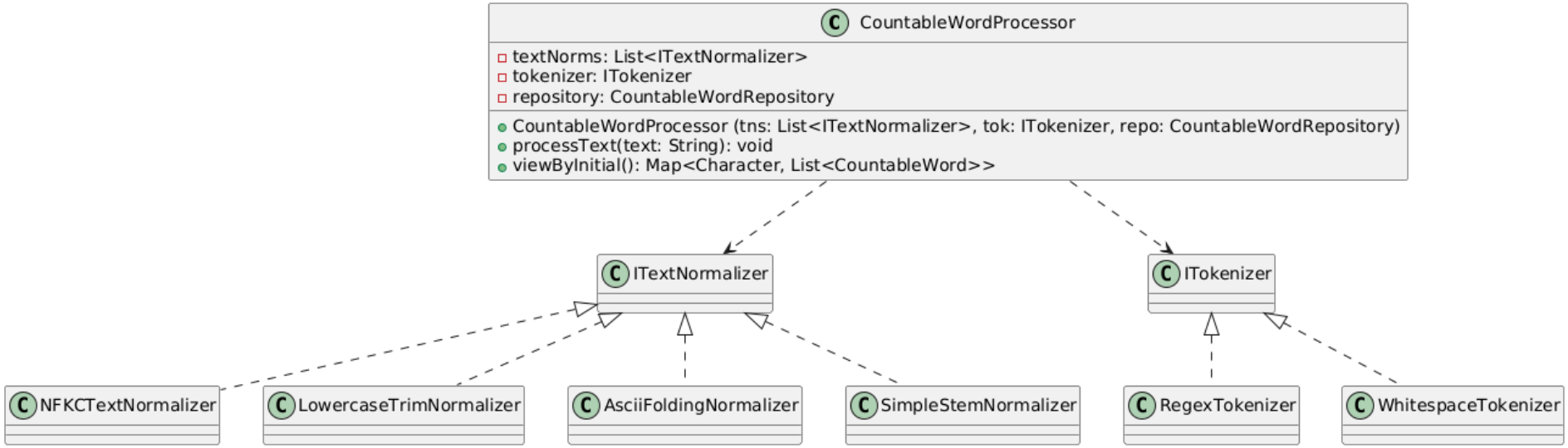
Lab2

- Strategy 클래스



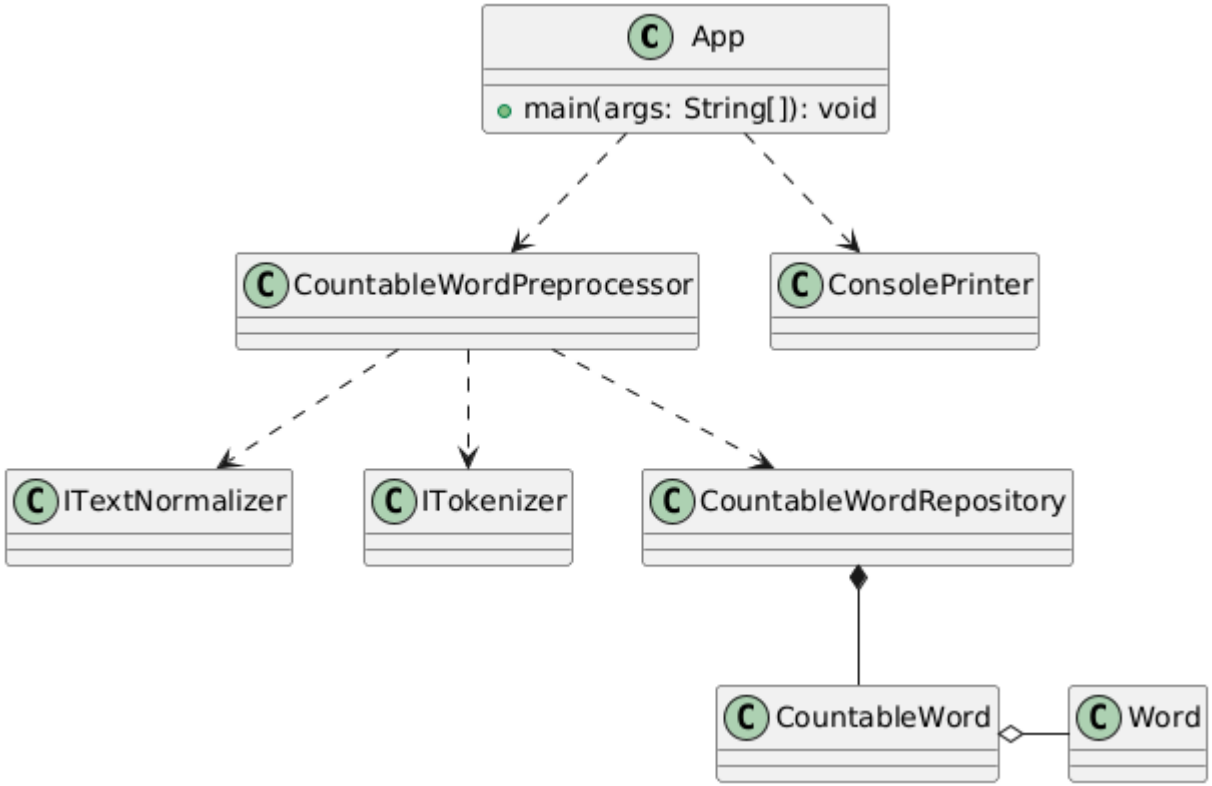
Lab2

- Strategy Pattern
  - CountableWordProcessor **processText** 에서는 1) **ITextNormalizer** 텍스트 정규화 전략과 2) **ITokenizer** 토큰화 전략을 사용하여 텍스트 처리 후 3) repository에 저장 및 4) 토큰 출력
  - CountableWordProcessor **viewByInitial**은



Lab2

- 전체 구성
  - App은 CountableWordProcessor에서 여러 전략을 사용하여 text 처리하여 CountableWordRepository에 넣어 초성별 빈도를 생성하고, 그 결과를 ConsolePrinter에서 출력



Lab2

```
public interface ITextNormalizer {
  String normalize(String raw);
}
public interface ITokenizer {
  List<String> tokenize(String text);
}
```

Lab2

```
import java.text.Normalizer;
// NFKC 유니코드 정규화
public class NFKCTextNormalizer implements
ITextNormalizer {
  @Override
  public String normalize(String raw) {
    if (raw == null) return "";
    return Normalizer.normalize(raw,
Normalizer.Form.NFKC);
  }
}
LowercaseTrimNormalizer // 공백제거 소문자화
AsciiFoldingNormalizer // 악센트/다이어크리틱 제거(é→e,
ö→o) 공백제거 소문자화
SimpleStemNormalizer // 소유격 복수형 시제/동명사 제거
```

Lab2

```
public class App {
  public static void main(String[] args) {
    String text = """
      Quick, brown   FOX!! jumps over 13 lazy dogs.
      QUICK—brown—fox;   (NFKC: “quotes”, full-width A B C, 
café, naïve)
      You're reading O'Reilly's book... Visit 
https://example.com/docs.
      Email: test@example.org, #hashtag @handle
      미나 프로그래밍—좋아요! 3 가지 항목:\tone, two, three.
      """;


..
```

Lab2

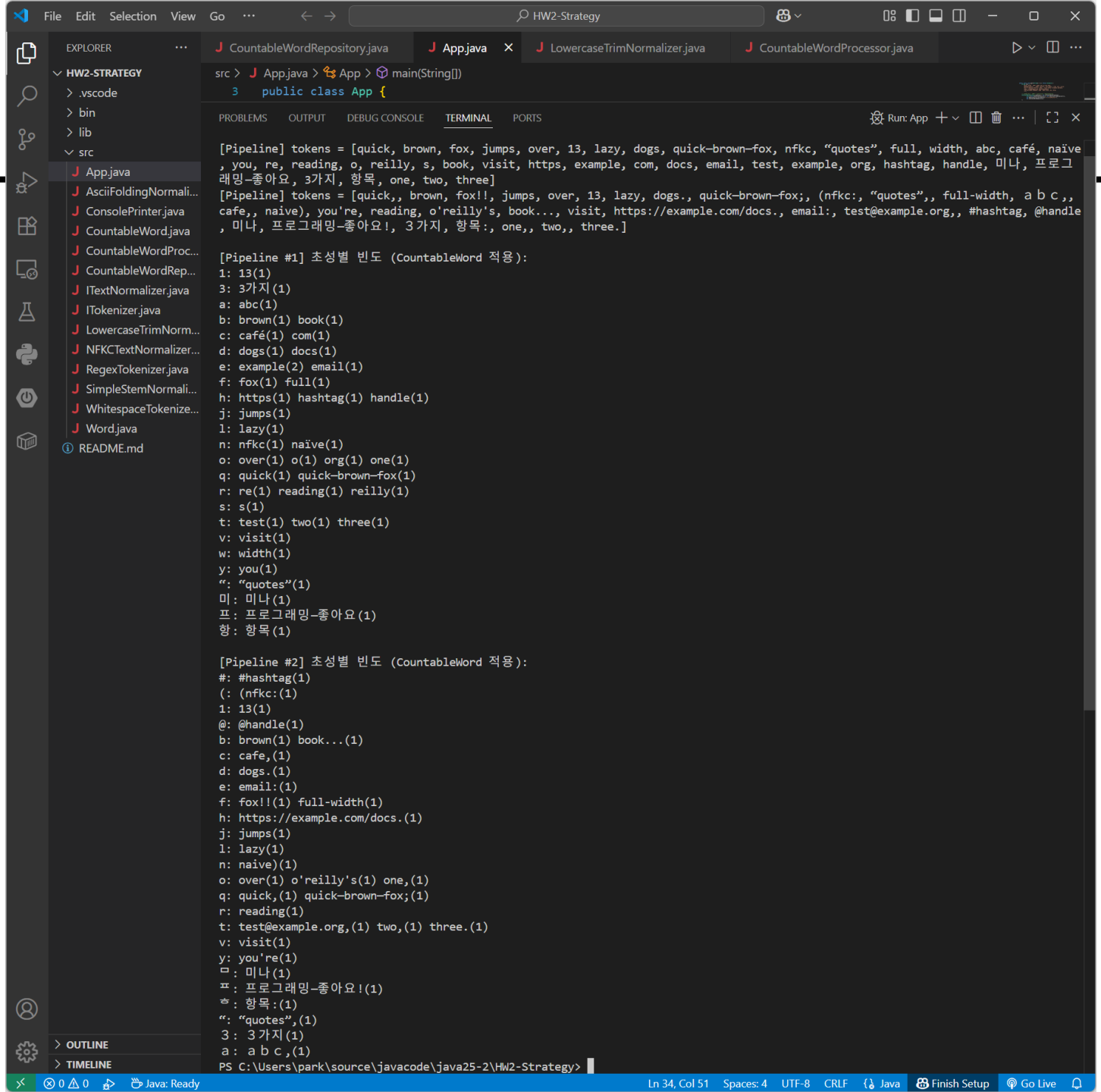
```
// 파이프라인 1: (NFKC → lower+trim) → RegexTokenizer
        Preprocessor pipeline1 = new Preprocessor(
                List.of(new NFKCTextNormalizer(), new
LowercaseTrimNormalizer()),
                new RegexTokenizer("[\\s\\p{Punct}]+"),
                new CountableWordRepository()
        );
// 파이프라인 2: (AsciiFolding -> SimpleStem) →
WhitespaceTokenizer
        Preprocessor pipeline2 = new Preprocessor(
                List.of(new NFKCTextNormalizer()),
                new WhitespaceTokenizer(),
                new CountableWordRepository()
        );
        pipeline1.processText(text);
        pipeline2.processText(text);
```

Lab2

```
// === CountableWord 적용: 파이프라인1 결과를 집계 ===
System.out.println("\n[Pipeline #1] 초성별 빈도 (CountableWord 적용):");
        new ConsolePrinter().print(pipeline1.viewByInitial());

        // === CountableWord 적용: 파이프라인2 결과를 집계 ===
System.out.println("\n[Pipeline #2] 초성별 빈도 (CountableWord 적용):");
        new ConsolePrinter().print(pipeline2.viewByInitial());
    } // end of main
} // end of App class
```

Lab2


File Edit Selection View Go
HW2-Strategy
EXPLORER
HW2-STRATEGY
.vscode
bin
lib
src
App.java
AsciiFoldingNormali...
ConsolePrinter.java
CountableWord.java
CountableWordProc...
CountableWordRep...
ITextNormalizer.java
ITokenizer.java
LowercaseTrimNorm...
NFKCTextNormalizer...
RegexTokenizer.java
SimpleStemNormali...
WhitespaceTokenize...
Word.java
README.md
CountableWordRepository.java
App.java
LowercaseTrimNormalizer.java
CountableWordProcessor.java
src > App.java > App > main(String[])
3 public class App {
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Run: App
[Pipeline] tokens = [quick, brown, fox, jumps, over, 13, lazy, dogs, quick-brown-fox, nfkc, "quotes", full, width, abc, café, naïve, you, re, reading, o, reilly, s, book, visit, https, example, com, docs, email, test, example, org, hashtag, handle, 미나, 프로그래밍-좋아요, 3가지, 항목, one, two, three]
[Pipeline] tokens = [quick,, brown, fox!!, jumps, over, 13, lazy, dogs., quick-brown-fox;, (nfkc:, "quotes",, full-width, ａｂｃ,, cafe,, naive), you're, reading, o'reilly's, book..., visit, https://example.com/docs., email:, test@example.org,, #hashtag, @handle, 미나, 프로그래밍-좋아요!, 3가지, 항목:, one,, two,, three.]
[Pipeline #1] 초성별 빈도 (CountableWord 적용):
1: 13(1)
3: 3가지(1)
a: abc(1)
b: brown(1) book(1)
c: café(1) com(1)
d: dogs(1) docs(1)
e: example(2) email(1)
f: fox(1) full(1)
h: https(1) hashtag(1) handle(1)
j: jumps(1)
l: lazy(1)
n: nfkc(1) naïve(1)
o: over(1) o(1) org(1) one(1)
q: quick(1) quick-brown-fox(1)
r: re(1) reading(1) reilly(1)
s: s(1)
t: test(1) two(1) three(1)
v: visit(1)
w: width(1)
y: you(1)
": "quotes"(1)
미: 미나(1)
프: 프로그래밍-좋아요(1)
항: 항목(1)
[Pipeline #2] 초성별 빈도 (CountableWord 적용):
#: #hashtag(1)
(: (nfkc:(1)
1: 13(1)
@: @handle(1)
b: brown(1) book...(1)
c: cafe,(1)
d: dogs.(1)
e: email:(1)
f: fox!!(1) full-width(1)
h: https://example.com/docs.(1)
j: jumps(1)
l: lazy(1)
n: naive)(1)
o: over(1) o'reilly's(1) one,(1)
q: quick,(1) quick-brown-fox;(1)
r: reading(1)
t: test@example.org,(1) two,(1) three.(1)
v: visit(1)
y: you're(1)
ㅁ: 미나(1)
ㅍ: 프로그래밍-좋아요!(1)
ㅎ: 항목:(1)
": "quotes",(1)
３: ３가지(1)
ａ: ａｂｃ,(1)
PS C:\Users\park\source\javacode\java25-2\HW2-Strategy>
OUTLINE
TIMELINE
Java: Ready
Ln 34, Col 51 Spaces: 4 UTF-8 CRLF Java Finish Setup Go Live

Submit to e-learning

- Lab2 과제에 yourcode (e.g.: 다른 strategy – CountableWordProcessor에 길이 기준 토큰 필터 적용 등)를 추가 (yourcode 없을시 10점에서 -1점 감점)
- **Java24-2-HW2-YourID-YourName.zip** 과제(보고서에 반드시 yourcode 설명 포함)를 e러닝에 제출 **(due by 9/23)**.