

LEC 01

CSE 122

File I/O

Questions during Class?

Raise hand or send here

sli.do #cse122



BEFORE WE START

Slido input & discuss with neighbors:

What is your favorite brand of
ice cream?

Instructor: Ivory Wang & Colin Lim

TAs: Dani
Rohan
Wesley
Connor
Yang

Lecture Outline (1/4)

- **Announcements** 
- Review Scanners for User Input
- Scanners for Files
 - Token-based, Line-based, and Hybrid processing
- Example

Announcements

- Hope you had fun in your first quiz section yesterday!
- Creative Project 0 (C0) releasing later today; due Thursday, July 2nd
 - Focused on Java Review + Functional Decomposition
- Office Hours Scheduled:
 - Tuesday 11:30am–12:30pm & Wednesday 12:00–1:00pm
 - Times don't work? Email us!
- IPL will also open on Monday, June 29th!
 - MGH 334, schedule is on the course website; staffed by our awesome TAs!
 - Open 1:30-5:30 most Mon-Thur, but check the schedule...
- Java Review Session Module Releasing on this Monday, June 29th

Lecture Outline (2/4)

- Announcements/Reminders
 - Review Java
- Review Scanners for User Input ◀
- Scanners for Files
 - Token-based, Line-based, and Hybrid processing
- Example

(Review) Scanner for User Input

Scanner is defined in the
java.util package

```
import java.util.*;
```

```
Scanner console = new Scanner(System.in);
```

Scanner Methods	Description
<code>nextInt()</code>	Reads the next token from the user as an <code>int</code> and returns it
<code>nextDouble()</code>	Reads the next token from the user as a <code>double</code> and returns it
<code>next()</code>	Reads the next token from the user as a <code>String</code> and returns it
<code>nextLine()</code>	Reads an <i>entire line</i> from the user as a <code>String</code> and returns it
<code>hasNextInt()</code>	Returns <code>true</code> if the next token can be read as an <code>int</code> , <code>false</code> otherwise
<code>hasNextDouble()</code>	Returns <code>true</code> if the next token can be read as a <code>double</code> , <code>false</code> otherwise
<code>hasNext()</code>	Returns <code>true</code> if there is another token of input to be read in, <code>false</code> otherwise
<code>hasNextLine()</code>	Returns <code>true</code> if there is another line of input to be read in, <code>false</code> otherwise

Lecture Outline (3/4)

- Announcements/Reminders
 - Review Java
- Review Scanners for User Input
- **Scanners for Files** 
 - Token-based, Line-based, and Hybrid processing
- Example

(PCM) Tokens

Tokens are units of input (as defined by the Scanner) that are separated by *whitespace* (spaces, tabs, new lines)



Practice : Think



sli.do #cse122

How many tokens are in the following line? (1/8)

“Hello world !” my-name is Dani

A) Four B) Five C) Six D) Seven



Practice : Pair



sli.do #cse122

How many tokens are in the following line? (2/8)

“Hello world !” my-name is Dani

A) Four B) Five C) Six D) Seven



Practice : Pair

[sli.do](#) [#cse122](#)

How many tokens are in the following line? (3/8)



“Hello world !” my-name is Dani

- A) Four** **B) Five** **C) Six** **D) Seven**



Practice : Pair



sli.do #cse122

How many tokens are in the following line? (4/8)



“Hello world !” my-name is Dani

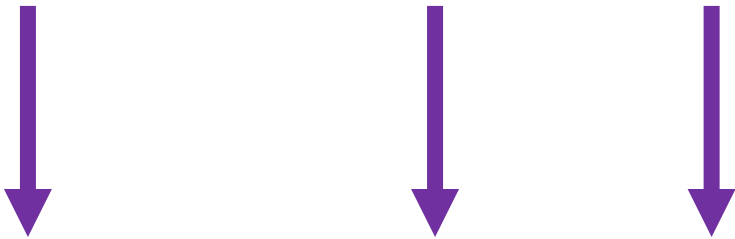
- A) Four B) Five C) Six D) Seven**



Practice : Pair

[sli.do](#) [#cse122](#)

How many tokens are in the following line? (5/8)



“Hello world !” my-name is Dani

- A) Four** **B) Five** **C) Six** **D) Seven**



Practice : Pair

[sli.do](#) [#cse122](#)

How many tokens are in the following line? (6/8)



“Hello world !” my-name is Dani

A) Four

B) Five

C) Six

D) Seven



Practice : Pair

[sli.do](#) [#cse122](#)

How many tokens are in the following line? (7/8)

↓ ↓ ↓ ↓ ↓
“Hello world !” my-name is Dani

- A) Four B) Five C) Six D) Seven**



Practice : Pair

[sli.do](#) [#cse122](#)

How many tokens are in the following line? (8/8)

↓ ↓ ↓ ↓ ↓ ↓

“Hello world !” my-name is Dani

- A) Four B) Five C) Six D) Seven

(PCM) Scanner for File I/O

Scanner is defined in the `java.util` package

```
import java.util.*;
```

File is defined in the `java.io` package

```
import java.io.*;
```

```
File file = new File("Example.txt");  
Scanner fileScan = new Scanner(file);
```

Scanner Methods	Description
<code>nextInt()</code>	Reads the next token as an <code>int</code> and returns it
<code>nextDouble()</code>	Reads the next token as a <code>double</code> and returns it
<code>next()</code>	Reads the next token as a <code>String</code> and returns it
<code>nextLine()</code>	Reads an <i>entire line</i> as a <code>String</code> and returns it
<code>hasNextInt()</code>	Returns <code>true</code> if the next token can be read as an <code>int</code> , <code>false</code> otherwise
<code>hasNextDouble()</code>	Returns <code>true</code> if the next token can be read as a <code>double</code> , <code>false</code> otherwise
<code>hasNext()</code>	Returns <code>true</code> if there is another token of input to be read in, <code>false</code> otherwise
<code>hasNextLine()</code>	Returns <code>true</code> if there is another line of input to be read in, <code>false</code> otherwise

(PCM) Typical Line Processing Pattern

```
while (scan.hasNextLine()) {  
    String nextLine = scan.nextLine();  
    // do something with nextLine  
}
```

(PCM) Typical Token Processing Pattern (1/2)

```
while (scan.hasNext____()) {  
    _____ nextToken = scan.next____();  
    // do something with nextToken  
}
```

(PCM) Typical Token Processing Pattern (2/2)

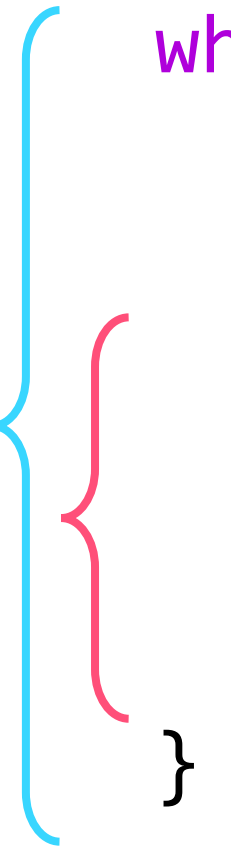
```
while (scan.hasNext____()) {  
    _____ nextToken = scan.next____();  
    // do something with nextToken  
}
```

Avoid mixing line- and token-based processing!

(PCM) Typical Hybrid Pattern (1/2)

```
while (fileScan.hasNextLine()) {  
    String line = fileScan.nextLine();  
    Scanner lineScan = new Scanner(line);  
    while (lineScan.hasNext__()) {  
        __ nextToken = lineScan.next__();  
        // do something with nextToken  
    }  
}
```

(PCM) Typical Hybrid Pattern (2/2)



```
while (fileScan.hasNextLine()) {  
    String line = fileScan.nextLine();  
    Scanner lineScan = new Scanner(line);  
    while (lineScan.hasNext__()) {  
        __ nextToken = lineScan.next__();  
        // do something with nextToken  
    }  
}
```

Line-by-line and then token-by-token to do something

(PCM) Scanners with Strings (not files!) (1/7)

```
String str = "A quick, brown fox";
```

```
Scanner stringScan = new Scanner(str);  
while (stringScan.hasNext____()) {  
    ____ nextToken = stringScan.next____();  
    // do something with nextToken  
}
```

(PCM) Scanners with Strings (not files!) (2/7)

```
String str = "A quick, brown fox";
```

```
Scanner stringScan = new Scanner(str);  
while (stringScan.hasNext()) {  
    String nextToken = stringScan.next();  
    System.out.println(nextToken);  
}
```

(PCM) Scanners with Strings (not files!) (3/7)

```
String str = "A quick, brown fox";
```

```
Scanner stringScan = new Scanner(str);  
while (stringScan.hasNext()) {  
    String nextToken = stringScan.next();  
    System.out.println(nextToken);  
}
```



(PCM) Scanners with Strings (not files!) (4/7)

```
String str = "A quick, brown fox";
```

```
Scanner stringScan = new Scanner(str);  
while (stringScan.hasNext()) {  
    String nextToken = stringScan.next();  
    System.out.println(nextToken);  
}
```

A

(PCM) Scanners with Strings (not files!) (5/7)

```
String str = "A quick, brown fox";
```

```
Scanner stringScan = new Scanner(str);  
while (stringScan.hasNext()) {  
    String nextToken = stringScan.next();  
    System.out.println(nextToken);  
}
```



quick,

(PCM) Scanners with Strings (not files!) (6/7)

```
String str = "A quick, brown fox";
```

```
Scanner stringScan = new Scanner(str);  
while (stringScan.hasNext()) {  
    String nextToken = stringScan.next();  
    System.out.println(nextToken);  
}
```



brown

(PCM) Scanners with Strings (not files!) (7/7)

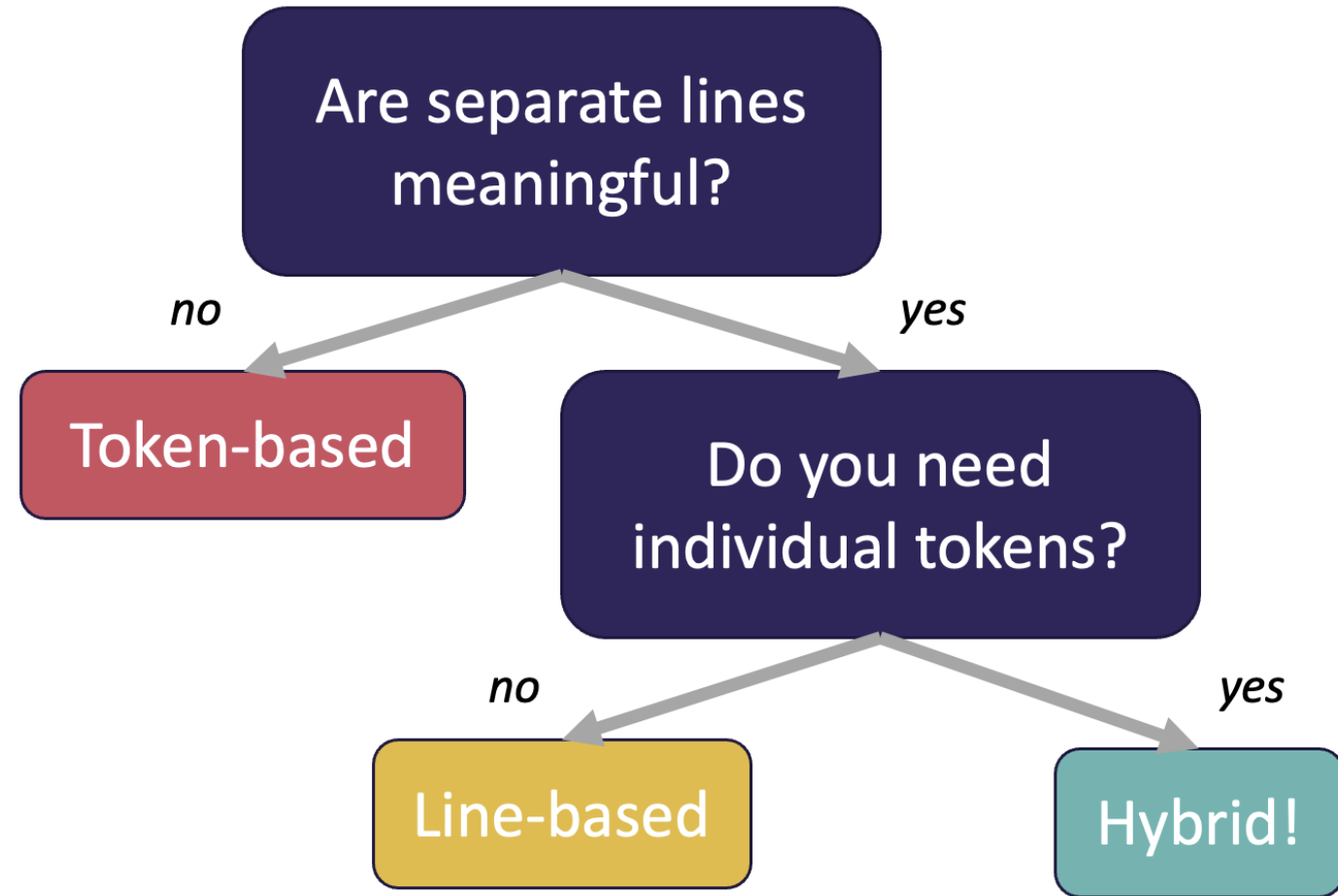
```
String str = "A quick, brown fox";
```

```
Scanner stringScan = new Scanner(str);  
while (stringScan.hasNext()) {  
    String nextToken = stringScan.next();  
    System.out.println(nextToken);  
}
```

fox

(PCM) Token vs. Line vs. Hybrid?

- We now know 3 different ways to use Files!
 - Line
 - Token
 - Hybrid
- Although this gives us flexibility – it can sometimes get confusing
- Feel free to use the following diagram to help!





Practice : Think



sli.do #cse122

Think: What is the output of this Java method?

```
File f = new File("Example.txt");
Scanner fileScan = new Scanner(f);
while (fileScan.hasNextLine()) {
    String line = fileScan.nextLine();
    Scanner lineScan = new Scanner(line);
    while (lineScan.hasNext()) {
        System.out.print(lineScan.next() + ", ");
    }
    System.out.println();
}
```

A) One, Two, Three,

B) One,
Two,
Three,

C) One, Two,
Three,

D) Error / Exception

Example.txt:

One Two
Three



Practice : Pair

[sli.do](#) [#cse122](#)

Pair: What is the output of this Java method?

```
File f = new File("Example.txt");
Scanner fileScan = new Scanner(f);
while (fileScan.hasNextLine()) {
    String line = fileScan.nextLine();
    Scanner lineScan = new Scanner(line);
    while (lineScan.hasNext()) {
        System.out.print(lineScan.next() + ", ");
    }
    System.out.println();
}
```

A) One, Two, Three,

B) One,
Two,
Three,

C) One, Two,
Three,

D) Error / Exception

Example.txt:

One Two
Three

Initial Letter Count by Line (1/2)

- Initial Letter Count
 - Token-based processing!
- Letter Count
 - Line-based processing!
- **Task: Initial Letter Count *by Line***
 - What kind of processing?

Initial Letter Count by Line (2/2)

- Initial Letter Count
 - Token-based processing!
- Letter Count
 - Line-based processing!
- **Task: Initial Letter Count *by Line***
 - What kind of processing?
 - Hybrid processing! Each line, and then each token per line.

Lecture Outline (4/4)

- Announcements/Reminders
 - Review Java
- Review Scanners for User Input
- Scanners for Files
 - Token-based, Line-based, and Hybrid processing
- Example ◀

Movie Ratings (1/2)

In this program, we'll be examining and printing data from a file of IMDb ratings for popular U.S. movies. This will happen through 3 major user-entered commands:

(F)ind movie, **(A)dd** a rating, and **(P)rint** ratings.

small.tsv

```
5
Title    Average Total
Pride_&_Prejudice  7.8 323647
Barbie   6.9 455488
Oppenheimer 8.4      588723
Poor_Things 8.5      20542
Spider-Man:_Across_the_Spider-Verse 8.6      329200
```

Movie Ratings (2/2)

Welcome to the CSE 122 Movie Rating Program!
Loaded 5 movies from small.tsv!

Menu: (F)ind movie, (A)dd rating, (P)rint ratings, (Q)uit

Enter your choice: F

What's the name of the movie? Pride & Prejudice

Movie Pride_&_Prejudice found!

Average Rating: 7.8

Total Ratings: 323647

Menu: (F)ind movie, (A)dd rating, (P)rint ratings, (Q)uit

Enter your choice: A

What movie would you like to add your rating to? Pride & Prejudice

And what rating would you like to give? 100000

Menu: (F)ind movie, (A)dd rating, (P)rint ratings, (Q)uit

Enter your choice: f

What's the name of the movie? Pride & Prejudice

Movie Pride_&_Prejudice found!

Average Rating: 8.1

Total Ratings: 323648

Menu: (F)ind movie, (A)dd rating, (P)rint ratings, (Q)uit

Enter your choice: P

5

Title Average Total

Pride_&_Prejudice 7.8 323647

Barbie 6.9 455488

Oppenheimer 8.4 588723

Poor_Things 8.5 20542

Spider-Man:_Across_the_Spider-Verse 8.6 329200

Menu: (F)ind movie, (A)dd rating, (P)rint ratings, (Q)uit

Enter your choice: q

Thank you for using this program! Bye!

small.tsv

5

Title Average Total

Pride_&_Prejudice 7.8 323647

Barbie 6.9 455488

Oppenheimer 8.4 588723

Poor_Things 8.5 20542

Spider-Man:_Across_the_Spider-Verse 8.6 329200



```
[Pride_&_..., Barbie, Oppenheimer, Poor_Things, Spider-Man...]
[7.8,          6.9,          8.4,          8.5,          8.6]
[323647,       455488,     588723,       20542,       329200]
```

Movie Ratings: Development Strategy

1. **Implement a method to load the movie rating data from the file and populate the appropriate arrays.**
2. Modify the print method to output to a file instead!
3. You are already given the rest of the program!

(PCM) PrintStreams for output

PrintStream is defined
in the `java.io` package

```
import java.io.*;
```

```
File outputFile = new File("out.txt");  
PrintStream output = new PrintStream(outputFile);
```

PrintStream Methods	Description
<code>print(...)</code>	Prints the given value to the set output location.
<code>println(...)</code>	Prints the given value to the set output location, and then terminates the line.

```
System.out.print("Hello, world! ");  
System.out.println("#1 Bee Movie fan!");
```

```
output.print("Hello, world! ");  
output.println("#1 Bee Movie fan!");
```

Hello, world! #1 Bee Movie fan!

Movie Ratings: Development Strategy

1. Implement a method to load the movie rating data from the file and populate the appropriate arrays. 
2. **Modify the print method to output to a file instead!**
3. You are already given the rest of the program!

(PCM) Checked Exceptions

If you try to compile a program working with file scanners, you may encounter this error message:

```
error: unreported exception FileNotFoundException; must be  
caught or declared to be thrown
```

To resolve this, you need to include `throws FileNotFoundException` at the end of the header of any method containing file Scanner creation code, or any method that calls that method!

This is like signing a waiver and telling Java—"Hey, I hereby promise to not get mad at you when you bug and crash my program if I give you a file that doesn't actually exist."

Movie Ratings

In this program, we'll be examining and altering data from a file of IMDb ratings for popular U.S. movies. This will happen through 3 major user-entered commands:

(F)ind movie, **(A)dd** a rating, and **(S)ave** to a file.

Instead of printing to console, we will now output to a file!

Movie Ratings

Welcome to the CSE 122 Movie Rating Program!
Loaded 5 movies from small.tsv!

Menu: (F)ind movie, (A)dd rating, (S)ave, (Q)uit
Enter your choice: F
What's the name of the movie? Pride & Prejudice
Movie Pride_&_Prejudice found!
Average Rating: 7.8
Total Ratings: 323647

Menu: (F)ind movie, (A)dd rating, (S)ave, (Q)uit
Enter your choice: A
What movie would you like to add your rating to? Pride & Prejudice
And what rating would you like to give? 100000

Menu: (F)ind movie, (A)dd rating, (S)ave, (Q)uit
Enter your choice: f
What's the name of the movie? Pride & Prejudice
Movie Pride_&_Prejudice found!
Average Rating: 8.1
Total Ratings: 323648

Menu: (F)ind movie, (A)dd rating, (S)ave, (Q)uit
Enter your choice: S
What's the name of the file you'd like to save to? out.txt

Menu: (F)ind movie, (A)dd rating, (S)ave, (Q)uit
Enter your choice: q
Thank you for using this program! Bye!

small.tsv

```
5
Title    Average Total
Pride_&_Prejudice  7.8 323647
Barbie    6.9 455488
Oppenheimer 8.4 588723
Poor_Things 8.5 20542
Spider-Man:_Across_the_Spider-Verse 8.6 329200
```



```
[Pride_&_..., Barbie, Oppenheimer, Poor_Things, Spider-Man...]
[7.8,          6.9,          8.4,          8.5,          8.6]
[323647,       455488, 588723,       20542,       329200]
```