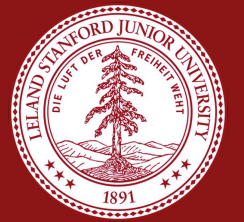
File Reading

Lecture 15

CS106A, Summer 2019
Sarai Gould && Laura Cruz-Albrecht

With inspiration from slides created by Keith Schwarz, Mehran Sahami, Eric Roberts, Stuart Reges, Chris Piech and others.



Announcements

- Midterm: check out website page
  - Download Bluebook, and be sure to have 2 factor authentication with passcodes; see Tuesday's lecture for helpful links.
- Midterm review session: tomorrow 10:30AM in Gates B01

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!
- Reminders about Due Dates and Late Days:

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!
- Reminders about Due Dates and Late Days:
  - One late day is 24 hours.

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!
- Reminders about Due Dates and Late Days:
  - One late day is 24 hours.
  - You have 3 late days for the whole quarter.

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!
- Reminders about Due Dates and Late Days:
  - One late day is 24 hours.
  - You have 3 late days for the whole quarter.
  - You can only use up to 2 late days on any assignment.

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!
- Reminders about Due Dates and Late Days:
  - One late day is 24 hours.
  - You have 3 late days for the whole quarter.
  - You can only use up to 2 late days on any assignment.
  - We will **not accept assignments more than 2 days late**. They will not be graded and will receive a zero.

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!
- Reminders about Due Dates and Late Days:
  - One late day is 24 hours.
  - You have 3 late days for the whole quarter.
  - You can only use up to 2 late days on any assignment.
  - We will **not accept assignments more than 2 days late**. They will not be graded and will receive a zero.
  - If you turn in an assignment late, but are out of late days, you will lose a bucket grade for style and functionality.

Announcements

- Assignment 3 was due at 10am!
- Assignment 4 will be out after lecture!
- Reminders about Due Dates and Late Days:
  - One late day is 24 hours.
  - You have 3 late days for the whole quarter.
  - You can only use up to 2 late days on any assignment.
  - We will **not accept assignments more than 2 days late**. They will not be graded and will receive a zero.
  - If you turn in an assignment late, but are out of late days, you will lose a bucket grade for style and functionality.
  - If your partner has late days and you don't, your personal grade will still go down if you both turn in the assignment late.

Plan for Today

- Review: Characters and Strings
- Reading Information from a File
- Reading Lines from a File
- Reading Strings from a File
- Practice: Let's Help Duke!

Review: Char Loops

```
// prints the characters a to z
for (char ch = 'a'; ch <= 'z'; ch++) {
    println(ch);
}
```

Review: Looping over Strings

A common String programming pattern is looping over a String and operating on each character.

```
for (int i = 0; i < str.length(); i++) {
    char ch = str.charAt(i);
    // do something with ch here
}
```

Review: Looping over Strings

Another common String programming pattern is **building up a new string** by adding characters to it over time.

```
// Creates a new String containing digits 0 through 4
String str = "";
for (int i = 0; i < 5; i++) {
    str += i;
}
println(str);  // 01234
```

Another Question

Uhhh, Sarai? I have a question.

Another Question

I got a high score in my breakout game, and I really wanted to save it, but when I closed my program, I lost my score! 😭
How can I save my high score?

How Can We Save Information?

How Can We Save Information?

In a Database

In a File

In our Mind?

How Can We Save Information?

~~In a Database~~

In a File

~~In our Mind?~~

Almost Every Program Uses Files!

Almost every program you use **reads files** from disk:

- Word processing (documents)
- Web browser (cookies)
- Games (saved progress)
- Eclipse (Java files)
- Music player (songs)
- And so many more!

What are Files?

Files are basically just a series of **bits** (zeros and ones).

Those bits can have different structures.

- Plain-Text: Bits represent characters.
- JPEG: Bits encode information about the structure of an image.
- MP3: Bits encode frequency information about music, etc.

What are Files?

Files are basically just a series of **bits** (zeros and ones).

Those bits can have different structures.

- Plain-Text: Bits represent characters.
- ~~JPEG: Bits encode information about the structure of an image.~~
- ~~MP3: Bits encode frequency information about music, etc.~~

Reading in a File

```
Are you there Diary? It’s me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

First, Let's Open the File!

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Opening a File:

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));
```

Importing Libraries

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Ooops, missing something!

```
import java.util.*; // for Scanner
import java.io.*; // for file


...
Scanner input = new Scanner(new File("Dukes Diary.txt"));
```

Now, Let's Read the File!

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Now, let's read in a line!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Now, let's read in a line!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

// "Are you there Diary? It's me, Duke."
String line1 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

The Scanner moves to the next line…

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

// "Are you there Diary? It's me, Duke."
String line1 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

We can continue reading Duke's Diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

// "Are you there Diary? It's me, Duke."
String line1 = input.nextLine();

// ""
String line2 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

We can continue reading Duke's Diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

// "Are you there Diary? It's me, Duke."
String line1 = input.nextLine();

// ""
String line2 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

We can continue reading Duke's Diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

...

// "I had a really tough day yesterday. Between stack"
String line3 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

We can continue reading Duke's Diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

...

// "I had a really tough day yesterday. Between stack"
String line3 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

We can continue reading Duke's Diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

...

// "I had a really tough day yesterday. Between stack"
String line3 = input.nextLine();

// "overflows and integer division, it felt like I got"
String line4 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

We can continue reading Duke's Diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

...

// "I had a really tough day yesterday. Between stack"
String line3 = input.nextLine();

// "overflows and integer division, it felt like I got"
String line4 = input.nextLine();
```

Now, Let's Read the File!


```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

We can continue reading Duke's Diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

...

// "nothing done! Hopefully today will be a better day!"
String line5 = input.nextLine();
```

What Should We Do Next?

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

What would you do if you just read someone's diary?

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

...

// "nothing done! Hopefully today will be a better day!"
String line5 = input.nextLine();
```

Closing the Scanner!

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Close the diary!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

...

// "nothing done! Hopefully today will be a better day!"
String line5 = input.nextLine();

input.close();
```

Simplifying Our Code

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Reading each line one at a time became tedious! And what if we don't know how many lines are in the Diary*?!?

Let's see if we can simplify our strategy.

*Errrr, I mean file

Reading One Line at a Time

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Let's use `hasNextLine()`!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));


while(input.hasNextLine()){
     // What should we do in here?

}
```

Reading One Line at a Time

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

As long as there is another line, let's print it!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

// This will print out every line in the file!
while(input.hasNextLine()){
     String line = input.nextLine();
     println(line);
}
```

Missing Something Again!

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Wait, what are we missing?

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

// This will print out every line in the file!
while(input.hasNextLine()){
     String line = input.nextLine();
     println(line);
}
```

Always Close the Scanner!

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

Make sure to close the Diary* when you're done with it!

```
Scanner input = new Scanner(new File("Dukes Diary.txt"));

// This will print out every line in the file!
while(input.hasNextLine()){
     String line = input.nextLine();
     println(line);
}
input.close();
```

*Definitely meant Scanner

Let’s Code It!

There's a Catch

```
Are you there Diary? It's me, Duke.

I had a really tough day yesterday. Between stack
overflows and integer division, it felt like I got
nothing done! Hopefully today will be a better day!
```

If we can't open the file, we need to **catch** the error.

```
try {
      Scanner input = new Scanner(new File("Dukes Diary.txt"));
      while(input.hasNextLine()){
            String line = input.nextLine();
            println(line);
      }
      input.close();
} catch (FileNotFoundException ex){
      println("Couldn't open Duke's Diary!");
}
```

There's a Catch

We will try to open the file, but if we can't it will throw a **`FileNotFoundException`**. We need to catch that error and print out a message.

```
try {
     // trying to open file!
     Scanner input = new Scanner(new File("filenameToOpen.txt"));
     ...
     input.close();
} catch (FileNotFoundException ex){
     println("Couldn't open this file!");
}
```

There's a Catch

We try to open the file...

```
try {
     // trying to open file!
     Scanner input = new Scanner(new File("filenameToOpen.txt"));
     ...
     input.close();
} catch (FileNotFoundException ex){
     println("Couldn't open this file!");
}
```

There's a Catch

It's not very effective…

I CANT OPEN THAT FILE
WHY ARE YOU MAKING ME
OPEN THE FILE I CAN'T
OPEN IT WHAT WERE YOU

```
try {
    // trying to open file!
    Scanner input = new Scanner(new File("filenameToOpen.txt"));
    ...
    input.close();
} catch (FileNotFoundException ex){
    println("Couldn't open this file!");
}
```

More like `FileNotFoundTemperTantrum`…

There's a Catch

It's not very effective…

A `FileNotFoundException` is thrown.

```
try {
     // trying to open file!
     Scanner input = new Scanner(new File("filenameToOpen.txt"));
     ...
     input.close();
} catch (FileNotFoundException ex){
     println("Couldn't open this file!");
}
```

There's a Catch

This mean we ***skip all of the code that would have happened after we opened the file***. We also won't close the Scanner because we couldn't open the file in the first place!

```
try {
     // trying to open file!
     Scanner input = new Scanner(new File("filenameToOpen.txt"));
     ...
     input.close();
} catch (FileNotFoundException ex){
     println("Couldn't open this file!");
}
```

There's a Catch

It's not very effective…

A `FileNotFoundException` is thrown.

```
try {
     // trying to open file!
     Scanner input = new Scanner(new File("filenameToOpen.txt"));
     ...
     input.close();
} catch (FileNotFoundException ex){
     println("Couldn't open this file!");
}
```

There's a Catch

A `FileNotFoundException` is thrown.

Our program prints: "Couldn't open this file!"

```
try {
     // trying to open file!
     Scanner input = new Scanner(new File("filenameToOpen.txt"));

     input.close();
} catch (FileNotFoundException ex){
     println("Couldn't open this file!");
}
```

A More General Version

An `IOException` is thrown.

Our program prints: "The error is: [type of error]"

```
try {
     // trying to open file!
     Scanner input = new Scanner(new File("filenameToOpen.txt"));

     input.close();
} catch (IOException ex){
     println("The error is: " + ex);
}
```

Try-Catch

If you want to open a file, you **must** have a try-catch.

Your program needs a guarantee that you will do something if you are unable to open your file!

A Few Other Exceptions

`NoSuchElementException`

- You read past the end of input

`InputMismatchException`

- You read the wrong type of token ("hi" as an `int`)

What About One Word at a Time?

Sometimes, we might want to read data a little bit differently.

Instead of a line at a time, we might want to read in smaller pieces of data!

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

It's hard to see, but this is a file containing Duke's passwords!

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

What should we do first?

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

Open the file! … with a try-catch.

```
try {
      Scanner input = new Scanner(new File("Dukes Passwords.txt"));









} catch (IOException ex){
      println("The error is: " + ex);
}
```

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

Let's read through the file… but how?

```
try {
      Scanner input = new Scanner(new File("Dukes Passwords.txt"));









} catch (IOException ex){
      println("The error is: " + ex);
}
```

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

`.next()` will read in a one word String at a time.

```
try {
      Scanner input = new Scanner(new File("Dukes Passwords.txt"));
      String password1 = input.next();
      // "123456"
      println(password1);



} catch (IOException ex){
      println("The error is: " + ex);
}
```

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

`.next()` will read in a one word String at a time.

```
try {
      Scanner input = new Scanner(new File("Dukes Passwords.txt"));
      String password1 = input.next();
      // "123456"
      println(password1);
      String password2 = input.next();
      // "password"
      println(password2);
} catch (IOException ex){
      println("The error is: " + ex);
}
```

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

Using `.hasNext()` will check for more one word Strings!

```
try {
      Scanner input = new Scanner(new File("Dukes Passwords.txt"));
      // Will print all passwords in the file!
      while(input.hasNext()){
            String password = input.next();
            println(password);
      }

} catch (IOException ex){
      println("The error is: " + ex);
}
```

One Word at a Time

```
123456 password qwerty iloveCS106A letmein
```

And then we have to close the Scanner for the password file!

```
try {
      Scanner input = new Scanner(new File("Dukes Passwords.txt"));
      // Will print all passwords in the file!
      while(input.hasNext()){
            String password = input.next();
            println(password);
      }
      input.close();
} catch (IOException ex){
      println("The error is: " + ex);
}
```

Let’s Code It!

Different Scanner Methods

Method	Description
`sc.nextLine()`	reads and returns a one-*line* `String` from the file
`sc.next()`	reads and returns a one-word `String` from the file
`sc.nextInt()`	reads and returns an `int` from the file
`sc.nextDouble()`	reads and returns a `double` from the file
`sc.hasNextLine()`	returns `true` if there are any more lines
`sc.hasNext()`	returns `true` if there are any more tokens
`sc.hasNextInt()`	returns `true` if there is a next token and it's an `int`
`sc.hasNextDouble()`	returns `true` if there is a next token and it's a `double`
`sc.close();`	should be called when done reading the file

Can We Combine Scanner Methods?

Can We Combine Scanner Methods?

Absolutely! Let's write a program that will count how many words are in each line of Duke's Diary entry!

Let’s Code It!

Reading from a String

A Scanner can read from a String as well!

```
String text = "Scanners can read from plain Strings too!"
Scanner input = new Scanner(text);

// Will print all of the words in the text!
while(input.hasNext()){
    String word = input.next();
    println(word);
}
input.close();
```

Choosing Files

What if we want to pick the file we open?

```
try {
      Scanner input = new Scanner(new File(“myFileName.txt”));
      while(input.hasNextLine()){
            String line = input.nextLine();
            println(line);
      }
      input.close();
} catch (IOException ex){
      println(“The error is: ” + ex);
}
```

Choosing Files

What if we want to pick the file we open?

```
// Gets filename from user
String filename = promptUserForFile("What file would you like to open?");

try {
      Scanner input = new Scanner(new File(filename));
      while(input.hasNextLine()){
            String line = input.nextLine();
            println(line);
      }
      input.close();
} catch (IOException ex){
      println("The error is: " + ex);
}
```

Choosing Files

What if we want to pick the file we open?

```
// Strategy 2
String filename = promptUserForFile("What file would you like to open?");

try {
      Scanner input = new Scanner(new File(filename));
      while(input.hasNextLine()){
            String line = input.nextLine();
            println(line);
      }
      input.close();
} catch (IOException ex){
      println("The error is: " + ex);
}
```

This automatically prompts a user for a new file if they mess up the filename!

Choosing Files

What if we want to pick the file we open?

```
// Strategy 2
String filename = promptUserForFile("What file would you like to open?");

try {
      Scanner input = new Scanner(new File(filename));
      while(input.hasNextLine()){
            String line = input.nextLine();
            println(line);
      }
      input.close();
} catch (IOException ex){
      println("The error is: " + ex);
}
```

Note: we changed this to the variable `filename`.

Secrets...

In some text files in Sarai's Eclipse, she's hidden some secrets. Can we find the correct files and print out what's inside of them?

Let’s Code It!

Sarai, my high score?

Helping Duke Save High Scores!

Note: Right now, we can't write to a file, be we've learned how to extract data from a file. If we assume we've already stored Breakout high scores in a file, how can we extract it?

Helping Duke Save High Scores!

What Format Should We Save Our Data In?

What's the Pseudocode for Extracting it?

Helping Duke Save High Scores!

What Format Should We Save Our Data In?

Each line will have:

Type:	`String`	`int`	`double`
Data:	`name`	`bricks`	`seconds`

What's the Pseudocode for Extracting it?

Helping Duke Save High Scores!

What Format Should We Save Our Data In?

Each line will have:

Type:	`String`	`int`	`double`
Data:	`name`	`bricks`	`seconds`

What's the Pseudocode for Extracting it?

```
try to open file
    While there is a new line:
        Extract name
        Extract number of bricks
        Extract number of seconds
        Print data
catch any errors
    print error message
```

Reminder: Different Scanner Methods

Method	Description
`sc.nextLine()`	reads and returns a one-*line* `String` from the file
`sc.next()`	reads and returns a one-word `String` from the file
`sc.nextInt()`	reads and returns an `int` from the file
`sc.nextDouble()`	reads and returns a `double` from the file
`sc.hasNextLine()`	returns `true` if there are any more lines
`sc.hasNext()`	returns `true` if there are any more tokens
`sc.hasNextInt()`	returns `true` if there is a next token and it's an `int`
`sc.hasNextDouble()`	returns `true` if there is a next token and it's a `double`
`sc.close();`	should be called when done reading the file

Helping Duke Save High Scores!

What Format Should We Save Our Data In?

Each line will have:

Type:	`String`	`int`	`double`
Data:	`name`	`bricks`	`seconds`

What's the Pseudocode for Extracting it?

```
try to open file
    While there is a new line:
        Extract name
        Extract number of bricks
        Extract number of seconds
        Print data
catch any errors
    print error message
```

Let’s Code It!

Plan for Today

- Review: Characters and Strings
- Reading Information from a File
- Reading Lines from a File
- Reading Strings from a File
- Practice: Let's Help Duke!