

Pond Log Files

Using text files



Where did the high score go?

PAUSE AND TRY

A game sets highScore to 900. You play again tomorrow. Is it still 900?

TALK WITH A PARTNER

1. Where does a variable live while the program runs?
2. What happens to it when the program ends?
3. Where could the 900 be kept so it survives?



Write your answer to question 3 before we talk.

By the end of class you can...

- 1 Explain why a file keeps data that a variable can't.
- 2 Open a file with File and Scanner, including the imports and throws IOException.
- 3 Read tokens with next, nextInt, nextDouble, and nextBoolean, and predict what each returns.
- 4 Read a whole file of any length with a hasNext loop, then close it.
- 5 Read whole lines with nextLine and cut them into pieces with split.

A large, white, rounded square containing the number '1' in a bold, black, sans-serif font.

1

PART 1

Opening a file

Your first file program

FIRSTREAD.JAVA

```
import java.io.File;
import java.io.IOException;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) throws IOException {
        Scanner input = new Scanner(new File("weather.txt"));
        int first = input.nextInt();
        input.close();
        System.out.println("First day: " + first);
    }
}
```

weather.txt

72 68 75 71

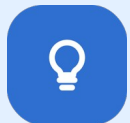
OUTPUT

First day: 72

File names the file.
Scanner reads from it.
close() says you are done.

Three things Java insists on

If you leave out...	Java says	When
<code>import java.io.File;</code>	<code>cannot find symbol</code>	Compiling
<code>throws IOException</code> on main	<code>unreported exception FileNotFoundException; must be caught or declared to be thrown</code>	Compiling
a file with that name	<code>FileNotFoundException</code> , and the program terminates	Running



Every method that opens a file, and every method that calls one, needs **throws IOException** in its header. `File` and `IOException` live in **java.io**.

One letter off, and the program stops



The compiler never checks the disk. A wrong file name **compiles**, then crashes when that line runs.

```
Scanner input = new Scanner(new File("wether.txt"));  
int first = input.nextInt();  
System.out.println(first);
```

In the folder: weather.txt

72 68 75 71

WHEN IT RUNS

```
Exception in thread "main" java.io.FileNotFoundException:  
wether.txt (No such file or directory)
```

A large, white, rounded square containing the number '2' in a bold, teal, sans-serif font.

2

PART 2

Reading tokens

The Scanner reading methods

Method	Returns	Throws InputMismatchException if...
<code>int nextInt()</code>	the next token as an int	it is not a whole number
<code>double nextDouble()</code>	the next token as a double	it is not a number
<code>boolean nextBoolean()</code>	the next token as a boolean	it is not true or false
<code>String next()</code>	the next token as a String	(any token works)
<code>String nextLine()</code>	the rest of the line, spaces and all	(any line works)
<code>boolean hasNext()</code>	true if a token is left to read	(it only looks)
<code>void close()</code>	nothing; closes the scanner	

Reading a record, token by token

player.txt

Mia 12 3.5
true

Call	Reads
next()	"Mia"
nextInt()	12
nextDouble()	3.5
nextBoolean()	true

```
Scanner input = new Scanner(new File("player.txt"));
String name = input.next();
int level = input.nextInt();
double speed = input.nextDouble();
boolean online = input.nextBoolean();
input.close();
System.out.println(name + " L" + level);
System.out.println(speed * 2);
System.out.println(online);
```

OUTPUT

```
Mia L12
7.0
true
```

Read in order

PAUSE AND TRY

Write the exact output. Bonus: what if the first read were `nextInt()`?

team.txt

Ana 3
Ben 7

```
Scanner input = new Scanner(new File("team.txt"));
String a = input.next();
int x = input.nextInt();
String b = input.next();
int y = input.nextInt();
input.close();
System.out.println(b + a);
System.out.println(x + y);
```



3

PART 3

Reading until the end

Keep reading while hasNext()

weather.txt

72 68 75 71



A **while** (**input.hasNext()**) loop reads a file of **any length**. Then call **close()**.

```
Scanner input = new Scanner(new File("weather.txt"));
int count = 0;
int sum = 0;
while (input.hasNext()) {
    sum += input.nextInt();
    count++;
}
input.close();
System.out.println(count + " days");
System.out.println((double) sum / count);
```

OUTPUT

```
4 days
71.5
```

Reading a fixed number of times



A loop that reads a set number of times assumes how big the file is. Read **past the end** and the program crashes.

WEATHER.TXT HOLDS 72 68 75 71

```
for (int i = 0; i < 5; i++) {  
    System.out.println(input.nextInt());  
}
```

OUTPUT

```
72  
68  
75  
71
```

THEN THE PROGRAM CRASHES

```
Exception in thread "main" java.util.NoSuchElementException
```

From a file into an array

grades.txt

5
88 92 75 100 81

OUTPUT

Best: 100

The first token is the **count**. It sizes the array.

```
Scanner input = new Scanner(new File("grades.txt"));
int[] grades = new int[input.nextInt()];
for (int i = 0; i < grades.length; i++) {
    grades[i] = input.nextInt();
}
input.close();
int best = grades[0];
for (int g : grades) {
    if (g > best) {
        best = g;
    }
}
System.out.println("Best: " + best);
```

A large, bold, teal number '4' is centered within a white rounded square that has a slight 3D effect and a shadow.

4

PART 4

Lines and split

Tokens or whole lines?

roster.txt

Grace Hopper
Alan Turing



Each loop below runs on its own new Scanner for roster.txt. Use **either** `nextLine` **or** the token readers on a file, not both.

NEXT()

```
while (input.hasNext()) {  
    String w = input.next();  
    System.out.println(w);  
}
```

OUTPUT

Grace
Hopper
Alan
Turing

NEXTLINE()

```
while (input.hasNext()) {  
    String line = input.nextLine();  
    System.out.println(line);  
}
```

OUTPUT

Grace Hopper
Alan Turing

KEY IDEA

split cuts a line at a delimiter

books.txt
Dune, Frank Herbert, 1965 Kindred, Octavia Butler, 1979

FIRST LINE, SPLIT ON ",",

parts.length is 3. Every piece is a **String**.

```
Scanner input = new Scanner(new File("books.txt"));
while (input.hasNext()) {
    String[] parts = input.nextLine().split(",");
    System.out.println(parts[0] + " by " + parts[1]);
}
input.close();
```

Index	Piece
0	"Dune"
1	"Frank Herbert"
2	"1965"

OUTPUT

Dune by Frank Herbert
Kindred by Octavia Butler

Predict the pieces

PAUSE AND TRY

Write what each line prints.

```
String[] t = "10:30:45".split(":");  
System.out.println(t.length);  
System.out.println(t[1]);  
System.out.println("a-b-c".split(" ").length);  
System.out.println(t[2] + 1);
```

YOUR OUTPUT

Words to know

File

Storage for data that persists when no program is running.

throws IOException

Required in the header of a method that opens a file, or calls one that does.

Token

A run of characters between whitespace, read by next, nextInt, and friends.

hasNext()

True if something is left to read. Use it as a while loop condition.

close()

Closes the scanner and its file. Call it after the last read.

split(del)

Returns a String array of the pieces between the delimiters.

Open, read, close



Explain to your partner, step by step, how a program reads every number in a file it has never seen.

SENTENCE STARTER

First I open it with _____, then I keep reading while _____, and at the end I _____.

Next: the Pond Log Files puzzle worksheet.