

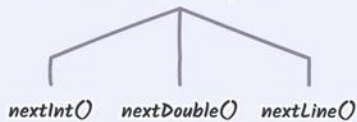


reading the keyboard, printing back, and keeping the result after the program ends

Reading input with Scanner

- `import java.util.Scanner;` brings the class in, and `Scanner input = new Scanner(System.in);` opens a channel to the keyboard
- `input.nextInt()` reads one int, `input.nextDouble()` reads one double, `input.next()` reads one word up to the next space
- `input.nextLine()` reads an entire line, including spaces, up to where Enter was pressed
- each next call waits for input before the next statement runs, so print a prompt with `System.out.print` before every read

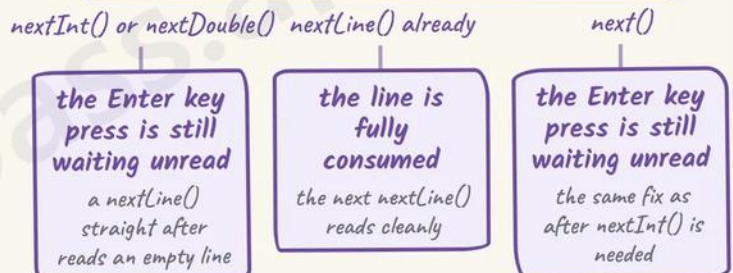
Scanner input



three read methods on the same Scanner object, chosen by what kind of value the input actually holds

The nextInt then nextLine trap

the previous read on this Scanner was



Fixing the nextInt then nextLine gap

- `nextInt()` reads only the digits, and leaves the Enter key press sitting unread in the input
- a `nextLine()` called straight after `nextInt()` reads that leftover Enter and returns an empty String, not the line the user meant to type
- an extra `input.nextLine();` with its result thrown away, placed right after the `nextInt()`, clears that leftover Enter
- this exact ordering is one of the most reliable questions asked on Scanner, since it looks correct and still returns the wrong value

A file's life in a program, in order

open

`new FileWriter` or
`new FileReader`
names the file

→
then

read or write

one line or one
piece of content
at a time

→
then

repeat

a loop continues
until the data or
the file is done

→
then

close

`close()` releases
the file, and
flushes anything
still buffered