Introducing the ScannerInput Class

Enhancing User Input Handling in Java

Produced by: Ms. Maireád Meagher
Ms Siobhan Roche

Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

Department of Computing and Mathematics
http://www.setu.ie/

Recap: Scanner for User Input

Scanner is used to read user input

- Example:
  - Scanner input = new Scanner(System.in);
  - int age = input.nextInt();

Problem: Buffer issues and invalid input handling

Common Problems with Scanner

Buffering issues when mixing nextInt() and nextLine()

No built-in input validation

Repetitive error-handling code in multiple places

Solution: ScannerInput Class

ENCAPSULATES INPUT LOGIC

ENSURES VALID DATA TYPES

ELIMINATES SCANNER BUFFER ISSUES

PROVIDES A REUSABLE INTERFACE

How ScannerInput Works

Example:

```
int age = ScannerInput.readNextInt("Enter your age: ");
```

How it works:

- Prompts user for input
- Parses input inside a loop
- Catches invalid values and retries

Key Methods in ScannerInput

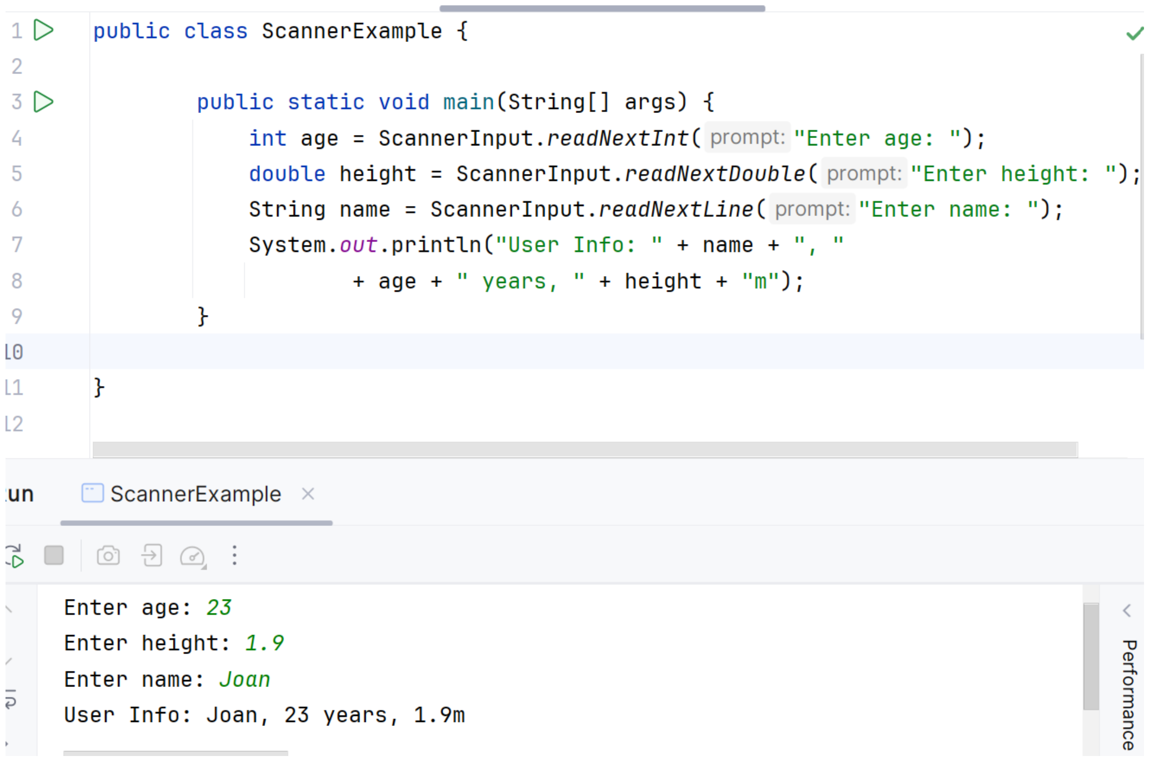
Method	Purpose
`readNextInt(String prompt)`	Ensures valid integer input
`readNextDouble(String prompt)`	Ensures valid double input
`readNextLine(String prompt)`	Reads a full string input
`readNextChar(String prompt)`	Reads a single character

Code Example

```
public class ScannerInput {

   /**
    * Read an int from the user.  If the entered data isn't actually an int,
    * the user is prompted again to enter the int.
    *
    * @param prompt  The information printed to the console for the user to read
    * @return The number read from the user and verified as an int.
    */
   public static int readNextInt(String prompt) {
      do {
         var scanner = new Scanner(System.in);
         try {
            System.out.print(prompt);
            return Integer.parseInt(scanner.next());
         }
         catch (NumberFormatException e) {
            System.err.println("\tEnter a number please.");
         }
      }  while (true);
   }
}
```

Example Usage of ScannerInput



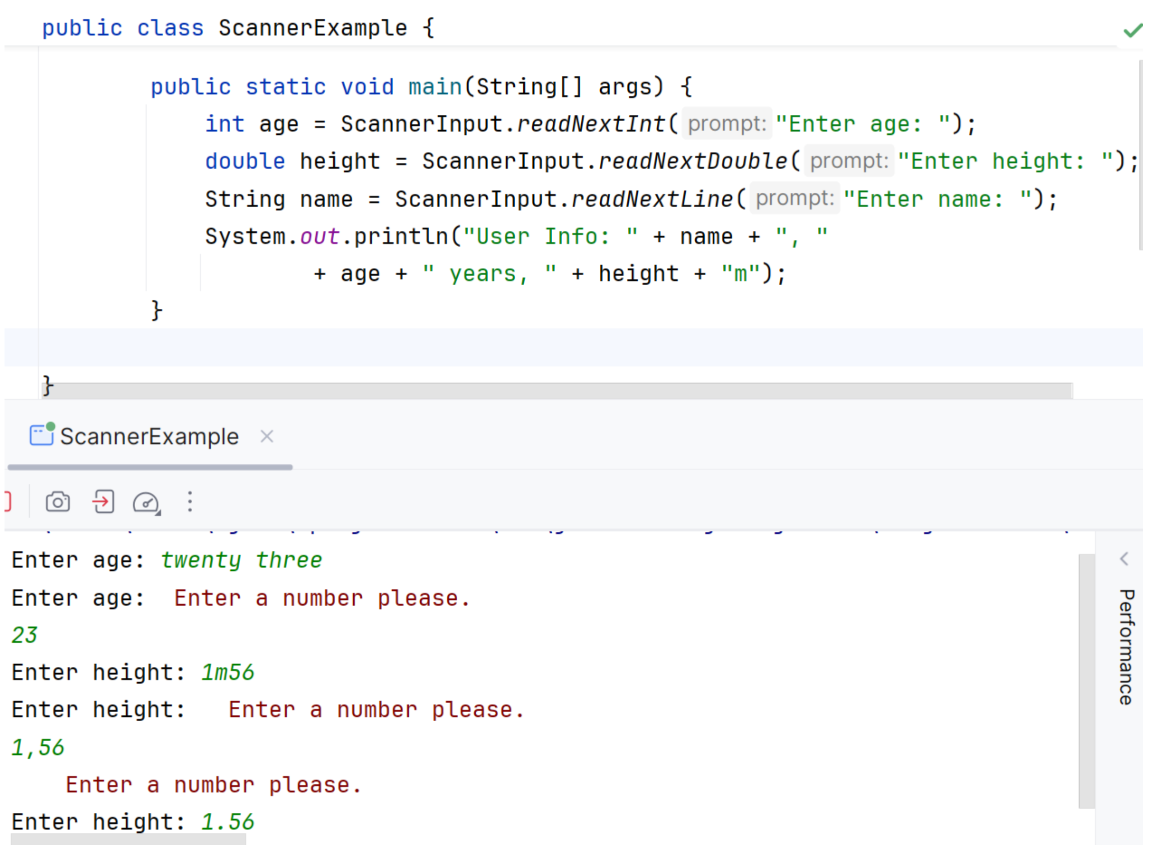
```java
public class ScannerExample {

        public static void main(String[] args) {
            int age = ScannerInput.readNextInt( prompt: "Enter age: ");
            double height = ScannerInput.readNextDouble( prompt: "Enter height: ");
            String name = ScannerInput.readNextLine( prompt: "Enter name: ");
            System.out.println("User Info: " + name + ", "
                    + age + " years, " + height + "m");
        }

}
```

```
Enter age: 23
Enter height: 1.9
Enter name: Joan
User Info: Joan, 23 years, 1.9m
```

Validation Example



```java
public class ScannerExample {

        public static void main(String[] args) {
            int age = ScannerInput.readNextInt( prompt: "Enter age: ");
            double height = ScannerInput.readNextDouble( prompt: "Enter height: ");
            String name = ScannerInput.readNextLine( prompt: "Enter name: ");
            System.out.println("User Info: " + name + ", "
                    + age + " years, " + height + "m");
        }

}
```

```
Enter age: twenty three
Enter age:  Enter a number please.
23
Enter height: 1m56
Enter height:   Enter a number please.
1,56
    Enter a number please.
Enter height: 1.56
```

Benefits of Using ScannerInput

- Prevents input errors (e.g., entering text instead of a number)
- Simplifies code (no need for try-catch everywhere)
- Reusable and efficient
- Improves user experience

Any Questions?