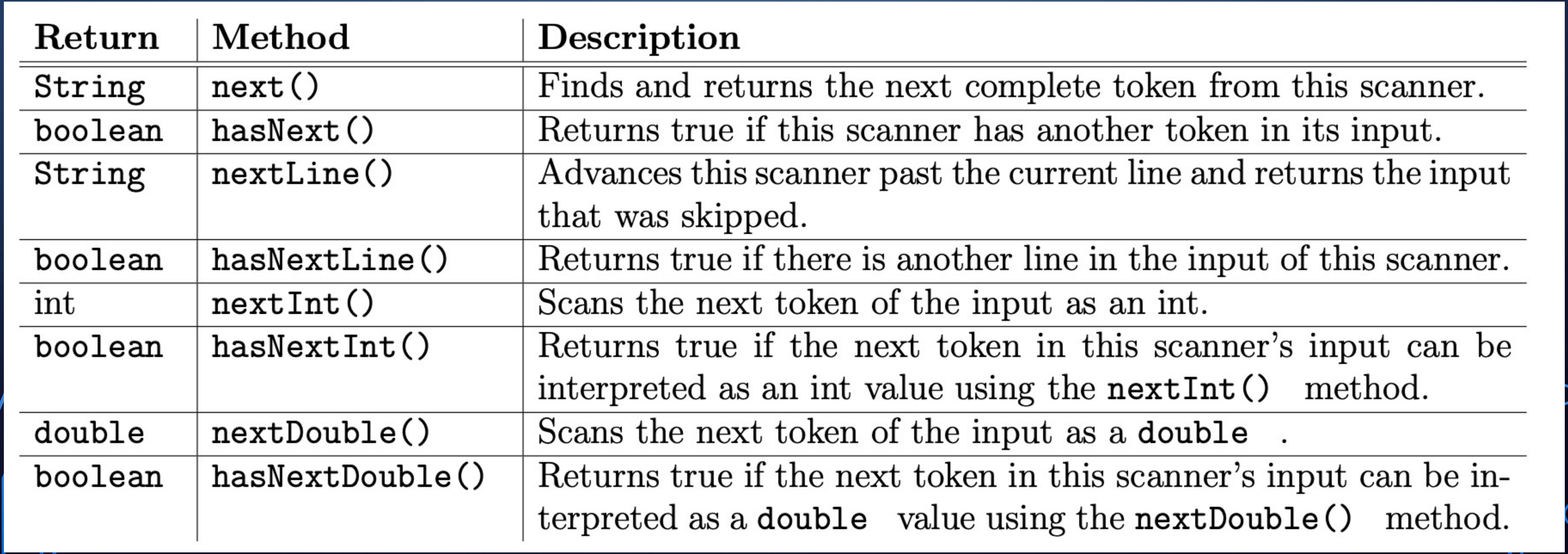
SCANNER HASNEXT METHODS

Java Scanner API

@ Dr. Jessica Young Schmidt and NCSU Computer Science Faculty

SCANNER HASNEXT METHODS

Each next() method has a corresponding hasNext() method.



Return	Method	Description
`String`	`next()`	Finds and returns the next complete token from this scanner.
`boolean`	`hasNext()`	Returns true if this scanner has another token in its input.
`String`	`nextLine()`	Advances this scanner past the current line and returns the input that was skipped.
`boolean`	`hasNextLine()`	Returns true if there is another line in the input of this scanner.
int	`nextInt()`	Scans the next token of the input as an int.
`boolean`	`hasNextInt()`	Returns true if the next token in this scanner's input can be interpreted as an int value using the `nextInt()` method.
`double`	`nextDouble()`	Scans the next token of the input as a `double` .
`boolean`	`hasNextDouble()`	Returns true if the next token in this scanner's input can be interpreted as a `double` value using the `nextDouble()` method.

ROBUST PROGRAMS

- Robustness
  - The degree to which erroneous situations are handled gracefully
- Want to write programs that execute when we present illegal data
  - Testing provides the illegal data
  - Now want to handle it

```java
import java.util.*;

/**
 * Allows user to examine how tokens are read.
 *
 * @author Jessica Young Schmidt
 */
public class ExamineInput {
  /**
   * Starts program
   *
   * @param args command line arguments
   */
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.print("Token? ");

    System.out.println("  hasNextInt = "
      + in.hasNextInt());
    System.out.println("  hasNextDouble = "
      + in.hasNextDouble());
    System.out.println("  hasNext = "
      + in.hasNext());
    System.out.println("  hasNextLine = "
      + in.hasNextLine());
  }
}
```

```
$ javac -d bin -cp bin src/ExamineInput.java

$ java -cp bin ExamineInput
Token? CSC116
  hasNextInt = false
  hasNextDouble = false
  hasNext = true
  hasNextLine = true

$ java -cp bin ExamineInput
Token? 3
  hasNextInt = true
  hasNextDouble = true
  hasNext = true
  hasNextLine = true

$ java -cp bin ExamineInput
Token? 11.6
  hasNextInt = false
  hasNextDouble = true
  hasNext = true
  hasNextLine = true

$ java -cp bin ExamineInput
Token? CSC 116
  hasNextInt = false
  hasNextDouble = false
  hasNext = true
  hasNextLine = true
```

```java
import java.util.Scanner;

public class RaceResults {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.print("Enter place (int): ");
    int place = in.nextInt();
    if (place == 1) {
      System.out.println("First Place!");
    } else if (place == 2) {
      System.out.println("Second Place!");
    } else if (place == 3) {
      System.out.println("Third Place!");
    } else {
      System.out.println("Finisher!");
    }
  }
}
```

```
$ javac -d bin -cp bin src/RaceResults.java

$ java -cp bin RaceResults
Enter place (int): 1
First Place!

$ java -cp bin RaceResults
Enter place (int): one
Exception in thread "main" java.util.InputMismatchException
        at java.base/java.util.Scanner.throwFor(Scanner.java:939)
        at java.base/java.util.Scanner.next(Scanner.java:1594)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
        at RaceResults.main(RaceResults.java:7)
```

HANDLING USER ERRORS

```java
import java.util.Scanner;

public class RaceResults {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.print("Enter place (int): ");

    // Check to see if the next value is an int. If
    // the next value is not an int, reprompt
    while (!in.hasNextInt()) {
      // Since we are within the while loop, we
      // know that the next value is not an int.
      // Therefore, we need to read in the next
      // value (as String) and reprompt for an int.
      in.next(); // discard input
      // Provide user with an error message and
      // reprompt for an int
      System.out.println("Not an int; try again.");
      System.out.print("Enter place (int): ");
    }

    // Now that we have made it past the while
    // loop, we know the next value
    // is an int. Therefore, we can read the next
    // value as an int.
    int place = in.nextInt();
    if (place == 1) {
      System.out.println("First Place!");
    } else if (place == 2) {
      System.out.println("Second Place!");
    } else if (place == 3) {
      System.out.println("Third Place!");
    } else {
      System.out.println("Finisher!");
    }
  }
}
```

```
$ javac -d bin -cp bin src/RaceResults.java

$ java -cp bin RaceResults
Enter place (int): 1
First Place!

$ java -cp bin RaceResults
Enter place (int): 10
Finisher!

$ java -cp bin RaceResults
Enter place (int): one
Not an int; try again.
Enter place (int): 4.5
Not an int; try again.
Enter place (int): two words
Not an int; try again.
Enter place (int): Not an int; try again.
Enter place (int): 5
Finisher!
```