COP-2210
Computer Programming I

Instructor: Dr. Antonio Hernandez

Text: Big Java: Early Objects, Interactive Edition, 6th Edition

The Java Language

14. Miscellanea

String versus char type

char:	primitive type
String:	object type

```
char c = 'a';
String s = "s";
```

Can only hold a single character at any time

String versus numerical types

numerical types (int, double, …):	primitive type
String:	object type

```
int i = 123;

String s = "123";
```

123 and **"123"** are different values!

Wrapper Classes

Wrapper class: a class that wraps the value of a primitive type into an object

Wrapper classes contain useful methods to be used in connection with the corresponding primitive types

Example

Integer class:

- *int parseInt (String s)* : returns an integer value represented by the String s

- *String toString (int i)* : returns a string representation of the int argument

Wrapper Classes

Class	Purpose
Boolean	Wraps a primitive type *boolean* into an object
Byte	Wraps a primitive type *byte* into an object
Character	Wraps a primitive type *char* into an object
Double	Wraps a primitive type *double* into an object
Float	Wraps a primitive type *float* into an object
Integer	Wraps a primitive type *integer* into an object
Long	Wraps a primitive type *long* into an object
Short	Wraps a primitive type *short* into an object

***Note*: All these classes are in java.lang**

Example: Using the Wrapper Classes

```
// Prog09_05 : The JOptionPane class
import javax.swing.*;
public class Prog09_05
{
    public static void main ( String args[ ] )
    {
            int n;
            String s;
            s = JOptionPane.showInputDialog ( null, "Enter a number",
                                "Input", JOptionPane.QUESTION_MESSAGE);
            n = Integer.parseInt(s);
            s = "This is the number you entered: " + n + "\n\n";

            JOptionPane.showMessageDialog ( null, s, "Output",
                                JOptionPane.INFORMATION_MESSAGE);
    }
}
```

Type Casting

Type casting: compiler conversion from one data type into another.

(<data type>) expression

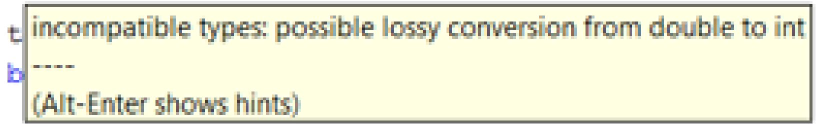
double x;

. . . $\underbrace{\textit{(int) x}}$ . . .

The type of this expression is **int**

Type Conversion: *Try it yourself*

```
// Program 14_02: type casting

import java.util.Scanner;

public class Prog14_02 {

    public static void main(String args[]){

        Scanner in = new Scanner(System.in);

        System.out.print("Enter a number with a decimal part: ");
        double d = in.nextDouble();

        int i = d;
        System.out.println("The number you entered, after casting, is: " + i);

    }
}
```



```
incompatible types: possible lossy conversion from double to int
----
(Alt-Enter shows hints)
```

Type Conversion: *Try it yourself*

```
// Program 14_03: type casting

import java.util.Scanner;

public class Prog14_03 {

    public static void main(String args[]){

        Scanner in = new Scanner(System.in);

        System.out.print("Enter a number with a decimal part: ");
        double d = in.nextDouble();

        int i = (int) d;
        System.out.println("The number you entered, after casting, is: " + i);

    }
}
```

OK

Reading Strings and Primitive Types: *Try it yourself*

```
import java.util.Scanner;
public class Prog14_04 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        System.out.print("Enter a number: ");
        int a = in.nextInt();

        in.nextLine();

        System.out.print("Enter a string: ");
        String str = in.nextLine();

        System.out.print("Enter a number: ");
        int b = in.nextInt();

        System.out.println("You entered: " + a + " " + str + " " + b);
    }
}
```

nextLine vs. next: *Try it yourself*

```
import java.util.Scanner;

public class Prog14_05
{
    public static void main(String[] args)
    {
        Scanner in = new Scanner(System.in);

        System.out.print("Enter two words separated by one space: ");
        String s1 = in.nextLine();
        System.out.println("You entered: " + s1);

        System.out.print("Enter two words separated by one space: ");
        String s2 = in.next();
        System.out.println("You entered: " + s2);
    }
}
```

nextLine(): reads input until the end of line

next(): reads input until first white space

Ctrl-space

Tip:

sout Ctrl-space: code completion shortcut available in NetBeans to have

System.out.println(“”)

completed for you.