CS112

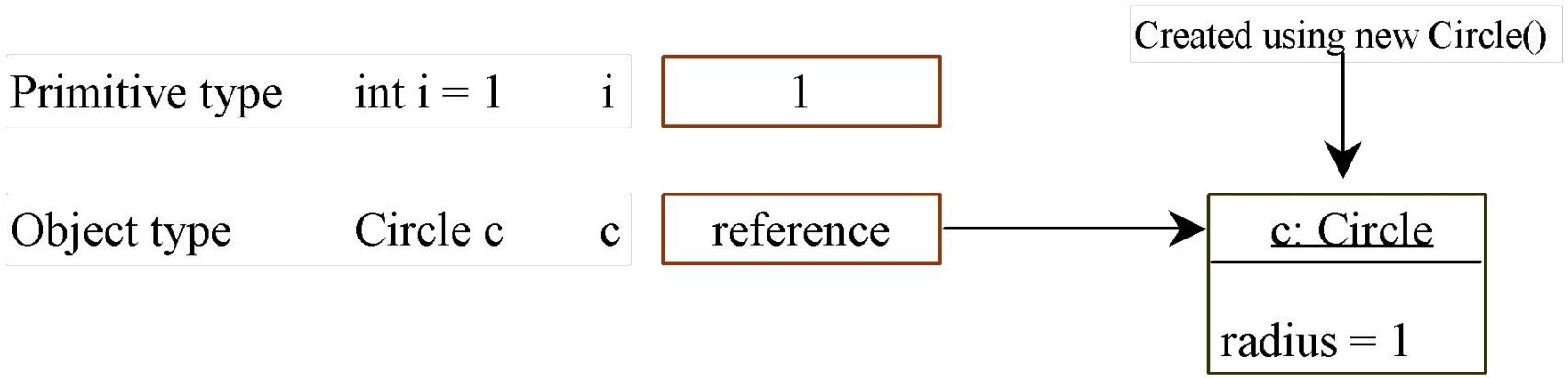
Objects and Classes (Part 2)

Lecture 03

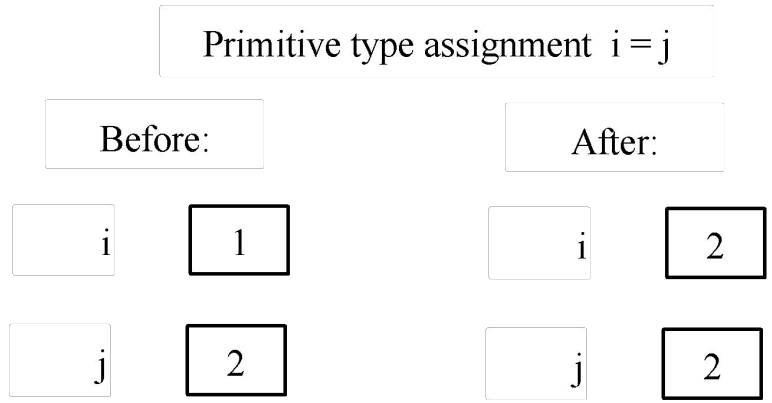
الفصل الدراسي الثاني 1442- Spring 2021

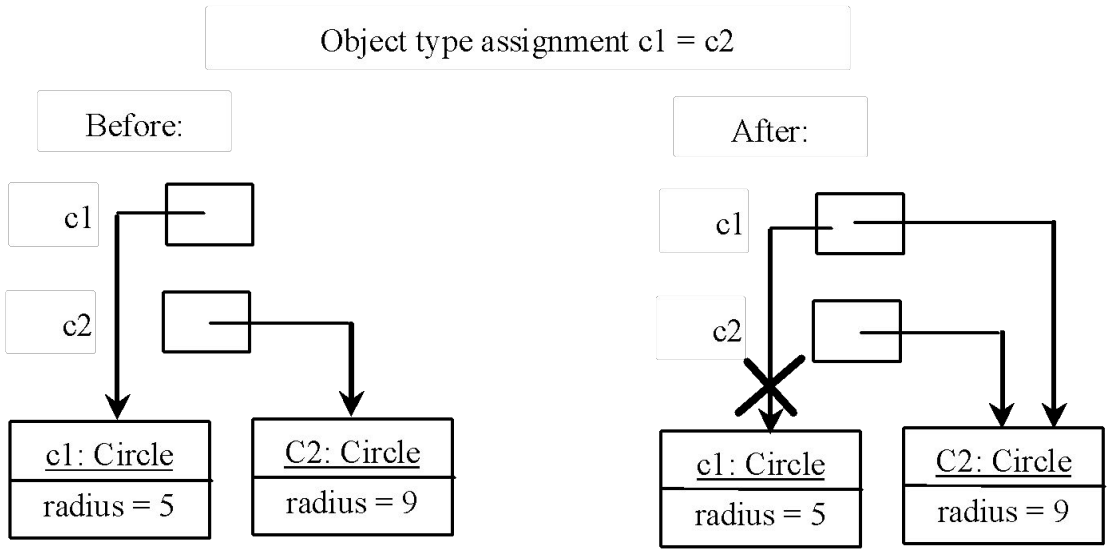
College of Computer Science and Engineering

Differences between Variables of Primitive Data Types and Object Types



Copying Variables of Primitive Data Types and Object Types



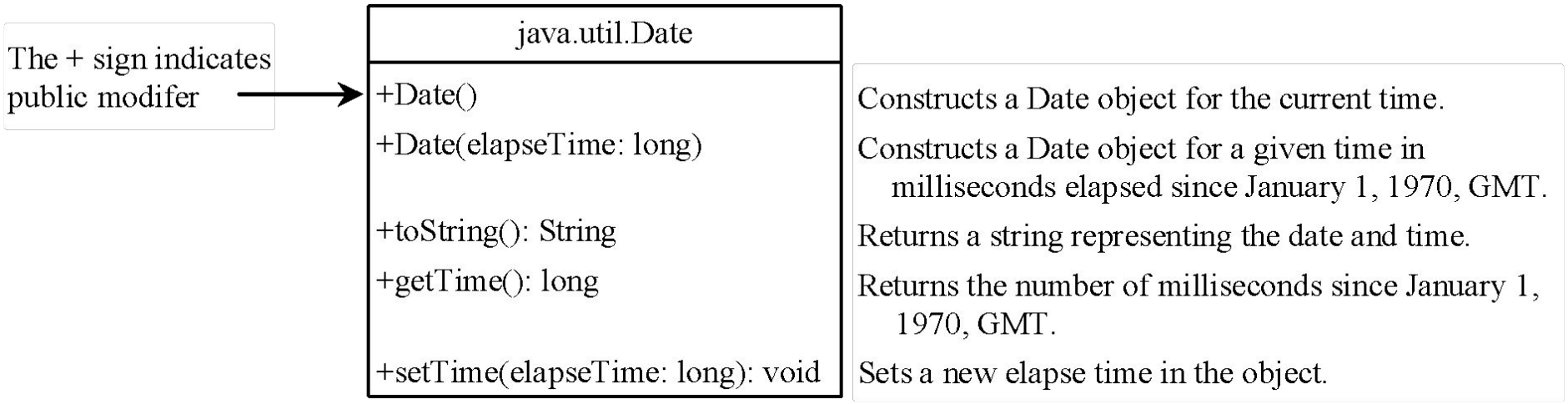


Garbage Collection

- As shown in the previous figure, after the assignment statement c1 = c2, c1 points to the same object referenced by c2.
- The object previously referenced by c1 is no longer referenced. This object is known as garbage. Garbage is automatically collected by JVM.
- TIP: If you know that an object is no longer needed, you can explicitly assign null to a reference variable for the object:
  - The JVM will automatically collect the space if the object is not referenced by any variable .

The Date Class

- Java provides a system-independent encapsulation of date and time in the java.util.Date class.
- You can use the Date class to create an instance for the current date and time and use its toString method to return the date and time as a string.



The Date Class Example

- For example, the following code

```
java.util.Date date = new java.util.Date();
System.out.println(date.toString());
```

displays a string like `Sun Mar 09 13:50:19 EST 2003`.

The Random Class

- You have used Math.random() to obtain a random double value between 0.0 and 1.0 (excluding 1.0). A more useful random number generator is provided in the java.util.Random class.

java.util.Random	
+Random()	Constructs a Random object with the current time as its seed.
+Random(seed: long)	Constructs a Random object with a specified seed.
+nextInt(): int	Returns a random int value.
+nextInt(n: int): int	Returns a random int value between 0 and n (exclusive).
+nextLong(): long	Returns a random long value.
+nextDouble(): double	Returns a random double value between 0.0 and 1.0 (exclusive).
+nextFloat(): float	Returns a random float value between 0.0F and 1.0F (exclusive).
+nextBoolean(): boolean	Returns a random boolean value.

The Random Class Example

- If two Random objects have the same seed, they will generate identical sequences of numbers.
- For example, the following code creates two Random objects with the same seed 3.

```
Random random1 = new Random(3);
System.out.print("From random1: ");
for (int i = 0; i < 10; i++)
  System.out.print(random1.nextInt(1000) + " ");
Random random2 = new Random(3);
System.out.print("\nFrom random2: ");
for (int i = 0; i < 10; i++)
  System.out.print(random2.nextInt(1000) + " ");
```

```
From random1: 734 660 210 581 128 202 549 564 459 961
From random2: 734 660 210 581 128 202 549 564 459 961
```

Instance Variables, and Methods

- Instance variables belong to a specific instance.
- Instance methods are invoked by an instance of the class.

Static Variables, Constants, and Methods (1)

- Static variables are shared by all the instances of the class.
- Static methods are not tied to a specific object.
- Static constants are final variables shared by all the instances of the class.

Static Variables, Constants, and Methods (2)

- To declare static variables, constants, and methods, use the static modifier.

Example 1 (1)

- This example adds a class variable numberOfObjects to track the number of Circle objects created.

```
public class CircleWithStaticMembers {
  /** The radius of the circle */
  double radius;

  /** The number of the objects created */
  static int numberOfObjects = 0;

  /** Construct a circle with radius 1 */
  CircleWithStaticMembers() {
    radius = 1.0;
    numberOfObjects++;
  }

  /** Construct a circle with a specified radius */
  CircleWithStaticMembers(double newRadius) {
    radius = newRadius;
    numberOfObjects++;
  }

  /** Return numberOfObjects */
  static int getNumberOfObjects() {
    return numberOfObjects;
  }

  /** Return the area of this circle */
  double getArea() {
    return radius * radius * Math.PI;
  }
}
```

Example 1 (2)

- This example adds a class variable numberOfObjects to track the number of Circle objects created.

```
public class TestCircleWithStaticMembers {
  /** Main method */
  public static void main(String[] args) {
    System.out.println("Before creating objects");
    System.out.println("The number of Circle objects is " +
      CircleWithStaticMembers.numberOfObjects);

    // Create c1
    CircleWithStaticMembers c1 = new CircleWithStaticMembers();

    // Display c1 BEFORE c2 is created
    System.out.println("\nAfter creating c1");
    System.out.println("c1: radius (" + c1.radius +
      ") and number of Circle objects (" +
      c1.numberOfObjects + ")");

    // Create c2
    CircleWithStaticMembers c2 = new CircleWithStaticMembers(5);

    // Modify c1
    c1.radius = 9;

    // Display c1 and c2 AFTER c2 was created
    System.out.println("\nAfter creating c2 and modifying c1");
    System.out.println("c1: radius (" + c1.radius +
      ") and number of Circle objects (" +
      c1.numberOfObjects + ")");
    System.out.println("c2: radius (" + c2.radius +
      ") and number of Circle objects (" +
      c2.numberOfObjects + ")");
  }
}
```

Example 1 (3)

- This example adds a class variable numberOfObjects to track the number of Circle objects created.