CISC 3115 TY2

Java API Classes: Date, Random, and Math

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Just discussed
  - Default constructor
  - Accessing objects via reference variables
  - Primitive and reference variables
  - Garbage collection
- A few classes in the Java Library (Java API)
  - Date, Random, Math

Java API and Library

- Java API: Java Application Programming Interface
  - Define the interface with which an application interacts with Java
    - Classes and methods that an application programmer can use in their own programs
- Java Library: implementation of the classes and methods

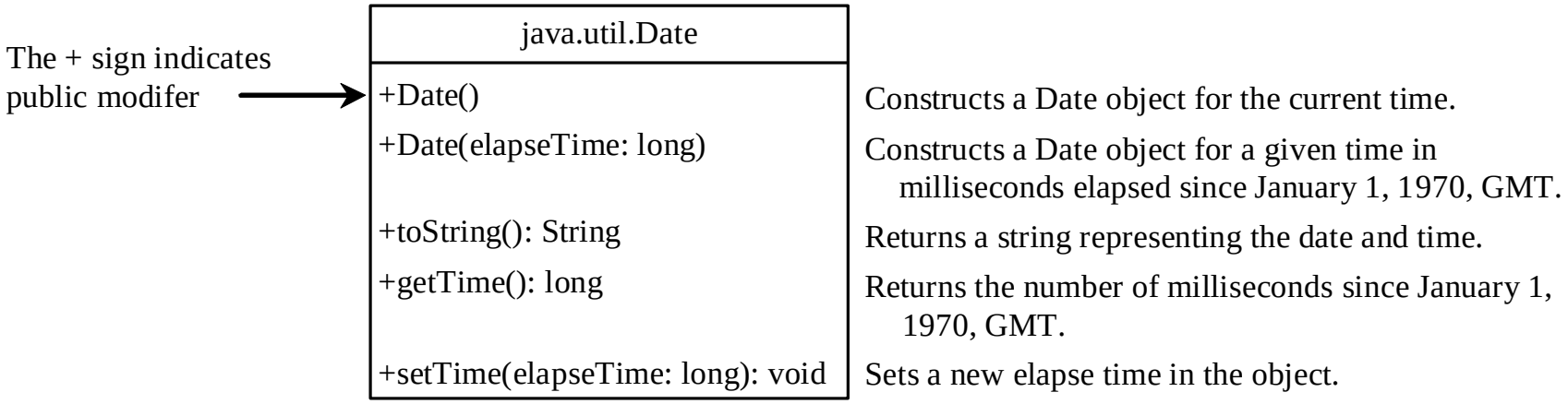
A Few Classes in Java Library

- Date
- Random
- Math

The Date Class

- A system-independent encapsulation of date and time in the java.util.Date class.
- Represent a specific instant in time, with millisecond precision
- Example usage:
  - 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.
- API documentation (may be intimidating to some, a good read nonetheless)
  - https://docs.oracle.com/javase/8/docs/api/java/util/Date.html

The Date Class: UML Class Diagram



- Why “public”?

The Date Class: Example

- Showing current system date and time
  - the import statement
  - Java package: java.util is a package while Date is a class in the package

PrintDate.java

```java
import java.util.Date;

public class PrintDate {
  public static void main(String[] args) {
    Date date = new Date();
    System.out.println(date.toString());

    long millis = date.getTime();
    System.out.println("It has been " + millis + " milliseconds since the Unix epoch");
  }
}
```

Deprecated Methods?

- A number of methods of the Date class are marked as deprecated.
- What does it mean?
- What should we do to get/set a specific Date object?
- Use the java.util.Calendar class or the java.time.LocateDateTime class

The Calendar Class

- A class provides methods for converting between a specific instant in time and a set of calendar fields such as YEAR, MONTH, DAY_OF_MONTH, HOUR, and so on.
- java.util.Calendar
- API documentation (may be intimidating to some, a good read nonetheless)
  - https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/util/Calendar.html

The Calendar Class: UML Class Diagram

java.util.Calendar
getInstance(): Calendar set(year:int, month:int, date:int, hourOfDay:int, minute:int, second:int): void setTime(date:Date): void getTime(): Date

The Calendar Class: Example

- Show a specific time and showing current system date and time

PrintCalendarDate.java

```java
import java.util.Date;
import java.util.Calendar;

public class PrintCalendarDate {
  public static void main(String[] args) {
    int year = 2023, month = 8, day = 7, hour=14, minute=15, second=12;
    Calendar calendar = Calendar.getInstance();
    calendar.set(year, month, day, hour, minute, second);
    Date date = calendar.getTime();
    System.out.println("The given date and time are " + date);

    long millis = date.getTime();
    System.out.println("It has been " + millis + " milliseconds since the Unix epoch");
  }
}
```

The LocaleDateTime Class

- Java provides a new system-independent encapsulation of date and time in the java.time.LocaleDateTime class.
- Represent a specific instant in time, with millisecond precision
- Common actions one would perform for a date
- API documentation (may be intimidating to some, a good read nonetheless)
  - https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/time/LocalDateTime.html

The LocaleDateTime Class: UML Class Diagram

java.time.LocaleDateTime
of(year:int, month:int, dayOfMonth:int, hour:int, minute:int, second:int):LocaleDateTime now(): LocaleDateTime

The LocaleDateTime Class: Example

- Show a specific time and showing current system date and time

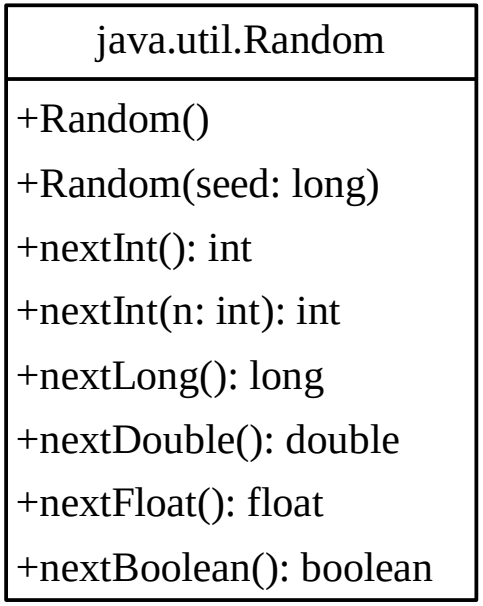
PrintLocalDateTime.java

```java
import java.time.LocalDateTime;
import java.time.ZoneOffset;

public class PrintLocalDateTime {
  public static void main(String[] args) {
    int year = 2023, month = 9, day = 7, hour=14, minute=15, second=12;
    LocalDateTime localDateTime = LocalDateTime.of(year, month, day, hour, minute, second);
    System.out.println("The given date and time are " + localDateTime);

    LocalDateTime localDateTimeNow = LocalDateTime.now();
    System.out.println("Now is " + localDateTimeNow);
    long millis = localDateTimeNow.toInstant(ZoneOffset.of("-4")).toEpochMilli();
    System.out.println("It has been " + millis + " milliseconds since the Unix epoch");
  }
}
```

The Random Class

- A pseudo-random number generator in the java.util.Random class
- Use it to generate a sequence of pseudo-random numbers
- Example usage:
  - Generate a sequence random integers
  - Generate a sequence random float point numbers
  - Generate a sequence random Boolean values
- API documentation (may be intimidating to some, a good read nonetheless)
  - https://docs.oracle.com/javase/8/docs/api/java/util/Random.html

The Random Class: UML Class Diagram



Constructs a Random object with the current time as its seed.

Constructs a Random object with a specified seed.

Returns a random int value.

Returns a random int value between 0 and n (exclusive).

Returns a random long value.

Returns a random double value between 0.0 and 1.0 (exclusive).

Returns a random float value between 0.0F and 1.0F (exclusive).

Returns a random boolean value.

The Random Class: Example

- What is a “seed”?

RandomNumbers.java

```java
import java.util.Random;

class RandomNumbers {
  public static void main(String[] args) {
    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) + " ");
    }

    Random random3 = new Random(4);
    System.out.print("\nFrom random3: ");
    for (int i = 0; i < 10; i++) {
      System.out.print(random3.nextInt(1000) + " ");
    }

    Random random4 = new Random();
    System.out.print("\nFrom random4: ");
    for (int i = 0; i < 10; i++) {
      System.out.print(random4.nextInt(1000) + " ");
    }

    Random random5 = new Random();
    System.out.print("\nFrom random5: ");
    for (int i = 0; i < 10; i++) {
      System.out.print(random5.nextInt(1000) + " ");
    }
  }
}
```

Pseudo-Random Numbers

- The Random class generates pseudo-random numbers, i.e., the numbers are generated by an algorithm
- Implication
  - They are in fact deterministic although appear random.
  - Given two identical seeds, the sequences of “random” numbers are identical as well

Math.random()

- The Math class in the Java Library has a random method
  - Generating pseudo-random double values in interval [0.0, 1.0)
  - Described in the API documentation
    - https://docs.oracle.com/javase/10/docs/api/java/lang/Math.html#random()

"When this method is first called, it creates a single new pseudorandom-number generator, exactly as if by the expression

```
new java.util.Random()
```

This new pseudorandom-number generator is used thereafter for all calls to this method and is used nowhere else. "

Questions?

- Concept of Java API and Java Library
- A few classes in the Java Library
  - Date and several related classes
  - Random
  - Math
- In which Java packages are they?