Programmazione Orientata agli Oggetti in Linguaggio Java

Tecniche di Programmazione: Date

versione 1.1

G. Mecca – Università della Basilicata – mecca@unibas.it

Sommario

- Date e Calendari
- Rappresentazione delle Date
- Utilizzo del Calendario
- Formattazione delle Date
- Nell'Applicazione

Date e Calendari

- Gestione appuntamenti
  - ⇨una delle funzionalità centrali è legata alla gestione di date e di orari
- In Java
  - ⇨la cosa è più complessa di quanto si pensi
- Perchè questa complessità ?
  - ⇨perchè Java è un linguaggio orientato all'internazionalizzazione

Date e Calendari

- Internazionalizzazione (I18N)
  - ⇨caratteristica del software di adattarsi automaticamente a zone del mondo e a contesti culturali diversi
  - ⇨richiede accortezze nella programmazione di dati sensibili al contesto geografico
  - ⇨e complica l'utilizzo delle API di Java
  - ⇨un esempio tipico sono le date

Date e Calendari

- La classe java.util.Locale
  - ⇨rappresenta la regione del mondo rispetto alla quale gestire l'internazionalizzazione
  - ⇨due informazioni fondamentali: la lingua (es: it, en ecc.) e il paese (es: it, us ecc.)
  - ⇨all'installazione viene creato un oggetto di tipo Locale che rappresenta la regione corrente
  - ⇨può essere cambiato nell'applicazione

Rappresentazione delle Date

- Il problema delle date
  - ⇨le date vengono rappresentate nel mondo in maniera completamente diversa
- Esempio: il giorno 13 giugno
  - ⇨non esiste nei paesi che non adottano il Calendario Gregoriano
  - ⇨esiste nel mondo occidentale, ma si chiama in ciascun paese in modo diverso (es: June, 13th)

Rappresentazione delle Date

ATTENZIONE
al concetto di "data" in Java

- L'approccio di Java
  - ⇨adottare un modo neutro per rappresentare le date
- Attenzione
  - ⇨una data di Java non corrisponde ad un giorno dell'anno, ma ad un istante di tempo
  - ⇨giorno, mese, anno, ore, minuti, secondi, millisecondi

Rappresentazione delle Date

- La classe java.util.Date
  - ⇨rappresenta una data (istante di tempo) come un valore in millisecondi a partire dal 1 gennaio 1970, ore 1:00
  - ⇨istanti precedenti corrispondono a valori negativi, date successive a valori positivi
  - ⇨il valore è di tipo long e consente di rappresentare un intervallo di date di 290 milioni di anni

Rappresentazione delle Date

- Alcuni esempi
  - java.util.Date d1 = new java.util.Date(0);
  - System.out.println(d1); // chiama toString()
  - java.util.Date d2 = new java.util.Date(86400000); //+ 1 giorno
  - System.out.println(d2);
  - java.util.Date d3 = new java.util.Date(-86400000); // -1 giorno
  - System.out.println(d3);
  - java.util.Date d = new java.util.Date(); // questo istante
  - System.out.println(d);

```
Thu Jan 01 01:00:00 CET 1970
Fri Jan 02 01:00:00 CET 1970
Wed Dec 31 01:00:00 CET 1969
Sun Jun 13 17:35:02 CEST 2004
```


Rappresentazione delle Date

- Cosa si può fare con un oggetto Date ?
  - praticamente niente
  - si tratta solo di un modo astratto per rappresentare gli istanti di tempo
- Le operazioni concrete con le date
  - creare un oggetto sulla base di giorno, mese ed anno e fare i conti con le date
  - stampare le date in vari formati

Rappresentazione delle Date

- Per manipolare giorni, mesi ed anni
  - ⇨serve un calendario che conosca giorni, mesi ed anni e consenta di fare i conti
- Per stampare le date
  - ⇨serve un “formattatore di date” che conosca i diversi formati delle diverse lingue
- Attenzione
  - ⇨questo vuol dire che per manipolare concretamente le date servono 3 oggetti (!)

Utilizzo del Calendario

- Il calendario
  - ⇨oggetto della classe java.util.GregorianCalendar
  - ⇨estende la classe astratta java.util.Calendar
  - ⇨per ora l’unica implementazione concreta di un calendario
  - ⇨consente di creare concretamente date
  - ⇨e di fare i conti con le date

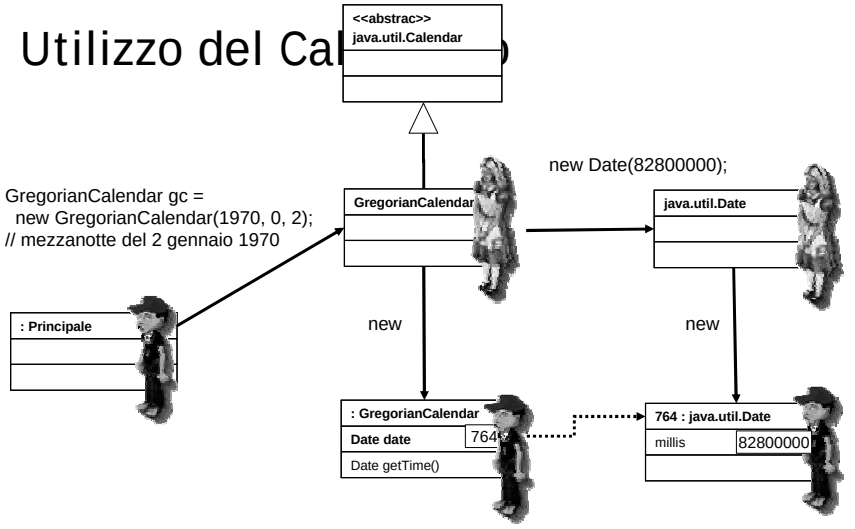
Utilizzo del Calendario

- I costruttori
  - GregorianCalendar() // questo istante
  - GregorianCalendar(int a, int m, int g)
  - GregorianCalendar(int a, int m, int g, int h, int mm)
- Attenzione
  - stiamo rappresentando istanti di tempo
  - fornendo solo giorno, mese ed anno si intende che ore e minuti coincidano con la mezzanotte
  - i mesi cominciano da 0 e non da 1

Utilizzo del Calendario

- Cosa succede chiamando il costruttore
  - viene creato un oggetto di tipo Date calcolando opportunamente i millisecondi
  - il calendario lavora in associazione con l'oggetto di tipo Date per svolgere i suoi compiti
  - è possibile eseguire il metodo Date getDate() per ottenere dal calendario il riferimento all'oggetto Date

Utilizzo del Cal



Utilizzo del Calendario

- Utilizzando l'oggetto di tipo calendario
  - ⇨è possibile manipolare concretamente la data
  - ⇨es: prelevare giorno, mese, anno, ore, minuti, secondi
  - ⇨es: confrontare date
  - ⇨es: sommare date

Utilizzo del Calendario

>> java.util.Calendar
>> java.util.GregorianCalendar

- I principali metodi di GregorianCalendar
  - il metodo get: passando un valore intero opportuno consente di ottenere giorno, mese, anno, ore, minuti, secondi
  - i valori interi corrispondono a costanti intere di java.util.Calendar
  - es: Calendar.YEAR, Calendar.MONTH
  - get(Calendar.YEAR) restituisce l'anno
  - get(Calendar.MONTH) restituisce il mese

Formattazione delle Date

- L'ultimo problema
  - stampare le date nella lingua e nel formato opportuno
- La classe java.text.DateFormat
  - consente di stampare le date in vari formati localizzati rispetto alla regione corrente
  - i formati: SHORT, MEDIUM, LONG e FULL

```
import java.util.Date;
import java.util.GregorianCalendar;
import java.util.Calendar;
import java.text.DateFormat;
import java.util.Locale;

public class Prova {

   public static void main(String[] args) {
      GregorianCalendar gc = new GregorianCalendar(2004, Calendar.JUNE, 14);
      Date d = gc.getTime();
      System.out.println("Millisecondi: " + d.getTime());
      DateFormat df = DateFormat.getDateInstance();
      System.out.println("Formato standard: " + df.format(d));
      DateFormat dfshort = DateFormat.getDateInstance(DateFormat.SHORT);
      System.out.println("Formato SHORT: " + dfshort.format(d));
      DateFormat dfmedium = DateFormat.getDateInstance(DateFormat.MEDIUM);
      System.out.println("Formato MEDIUM: " + dfmedium.format(d));
      DateFormat dflong = DateFormat.getDateInstance(DateFormat.LONG);
      System.out.println("Formato LONG: " + dflong.format(d));
      DateFormat dffull = DateFormat.getDateInstance(DateFormat.FULL);
      System.out.println("Formato FULL: " + dffull.format(d));
      DateFormat dffullfrancia = DateFormat.getDateInstance(DateFormat.FULL, Locale.FRANCE);
      System.out.println("Formato FULL Francese: " + dffullfrancia.format(d));
   }
}
```

```
Millisecondi: 1087164000000
Formato standard: 14-giu-2004
Formato SHORT: 14/06/04
Formato MEDIUM: 14-giu-2004
Formato LONG: 14 giugno 2004
Formato FULL: lunedì 14 giugno 2004
Formato FULL Francese: lundi 14 juin 2004
```


Nell'Applicazione

- Nell'applicazione Gestione Appuntamenti
  - ⇨una classe Giorno per rappresentare i giorni >> utilizza una data (calendario) con ore e minuti fissati convenzionalmente alle 00:00
  - ⇨una classe Orario per rappresentare gli orari >> utilizza una data (calendario) con giorno fissato convenzionalmente al 1 gennaio 1970

```
package it.unibas.appuntamenti.modello;

public class Giorno implements Comparable {

   private java.util.GregorianCalendar calendar;
   private java.util.List listaImpegni = new java.util.LinkedList();

   public Giorno(int gg, int mese, int anno) {
      if (gg < 0 || gg > 31) { throw new IllegalArgumentException("Giorno scorretto: " + gg); }
      if (mese < 1 || mese > 12) { throw new IllegalArgumentException("Mese scorretto: " + mese); }
      if (anno < 0) { throw new IllegalArgumentException("Anno scorretto: " + anno); }
      this.calendar = new java.util.GregorianCalendar(anno, mese - 1, gg);
   }

   public int getNumGiorno() { return this.calendar.get(java.util.Calendar.DAY_OF_MONTH); }
   public int getMese() { return this.calendar.get(java.util.Calendar.MONTH); }
   public int getAnno() { return this.calendar.get(java.util.Calendar.YEAR); }
   public int getGiornoSettimana() { return this.calendar.get(java.util.Calendar.DAY_OF_WEEK); }

   public String toShortString() {
      java.util.Date date = this.calendar.getTime();
      java.text.DateFormat dateFormat =
      java.text.DateFormat.getDateInstance(java.text.DateFormat.SHORT);
      return dateFormat.format(date);
   }
   ...
```


```
package it.unibas.appuntamenti.modello;

public class Orario implements Comparable {

   private java.util.GregorianCalendar calendar;

   public Orario(int ora, int minuti) {
      if (ora < 0 || ora > 23) { throw new IllegalArgumentException("Ora scorretta: " + ora); }
      if (minuti < 0 || minuti > 59) { throw new IllegalArgumentException("Minuti scorretti: " + minuti); }
      this.calendar =
            new java.util.GregorianCalendar(1970, java.util.Calendar.JANUARY, 1, ora, minuti);
   }

   public int getOra() { return this.calendar.get(java.util.Calendar.HOUR_OF_DAY); }

   public int getMinuti() { return this.calendar.get(java.util.Calendar.MINUTE); }

   public String toString() {
      String padOra = "";
      String padMinuti = "";
      if (this.getOra() < 10) { padOra = " "; }
      if (this.getMinuti() < 10) { padMinuti = "0"; }
      return padOra + getOra() + ":" + padMinuti + getMinuti();
   }
   ...
```

Riassumendo

- Date e Calendari
- Rappresentazione delle Date
- Utilizzo del Calendario
- Formattazione delle Date
- Nell'Applicazione

Termini della Licenza