Programmazione Orientata agli Oggetti in Linguaggio Java

Classi e Oggetti: C#

versione 2.1

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

Sommario

- Introduzione
- Alcune Differenze
  - ⇨Namespace
  - ⇨Convenzioni di Stile
  - ⇨La Classe Console
- Compilatore di C#
- Gli Esempi
- Documentazione di .NET

Introduzione

- Java e C#
  - ⇨i due linguaggi sono molto simili
- In particolare
  - ⇨entrambi sono orientati agli oggetti
  - ⇨entrambi discendono dal C++
  - ⇨C# è chiaramente ispirato a Java sia nella sintassi che nella semantica

Introduzione

- Ma
  - ⇨C# è un linguaggio più complesso di Java
- L'approccio alla presentazione di C#
  - ⇨per differenza rispetto a Java
  - ⇨trascurando alcune delle caratteristiche di C# ereditate dal C++ e non presenti in Java
  - ⇨es: gestione diretta della memoria

Introduzione

>> calcolatricestatica
>> calcolatrice

- C# è un linguaggio a oggetti
  - ⇨un'applicazione C# è fatta di componenti
  - ⇨i componenti hanno proprietà e metodi
  - ⇨due tipi di componenti: classi e oggetti
  - ⇨gli oggetti sono creati dalle classi usando i costruttori
  - ⇨e si manipolano utilizzando i riferimenti

Alcune Differenze

- Per cominciare
  - ⇨riassumiamo le principali differenze tra i due linguaggi
- In particolare
  - ⇨tipi di dato
  - ⇨namespace
  - ⇨convenzioni di stile
  - ⇨utilizzo della console

Alcune Differenze

- Principali differenze
  - il tipo dei booleani si chiama bool (come in C++)
  - esiste un tipo di base per le stringhe, il tipo string
  - ma esiste anche una classe per le stringhe, System.String; i due tipi sono intercambiabili
  - le costanti si dichiarano con const e non con static final (const = static + final)

Namespace

- Organizzazione dei componenti in .NET
  - attraverso i namespace
- Namespace (Spazio di Nomi)
  - strumento per qualificare i nomi delle classi
  - nome completo di una classe: <nameSpace>.<nomeClasse>
- Esempi
  - System.Console, System.IO.File

Namespace

- Attenzione alla differenza
  - ⇨System in java è una classe del package java.lang
  - ⇨System in C# è il namespace principale (il corrispondente di java.lang)
- A cosa è dovuta la confusione
  - ⇨alle diverse convenzioni di stile
  - ⇨i namespace di C# cominciano con la maiuscola (>>)

Namespace

ATTENZIONE
alla differenza tra package e namespace

- Differenze con i package di Java
  - ⇨un package di Java è un namespace ma anche ad una cartella del disco
  - ⇨viceversa, in C# non c'è nessun legame tra la struttura del codice e la struttura dei file
- Ovvero
  - ⇨le classi di un namespace possono stare in qualsiasi cartella del disco

Namespace

- Inoltre
  - il codice di una classe che si chiama Prova può stare in qualsiasi file con estensione .cs
  - non necessariamente Prova.cs
- Sintassi per dichiarare il namespace
  - namespace <nome> { <codice della classe> }
  - attenzione alle parentesi graffe
  - es: namespace Unibas.Utilita { // classe Console }

Namespace

- Per utilizzare un namespace
  - direttiva using
  - analoga ad import, con una differenza (>>)
  - es: using System;
  - consente di scrivere nel codice Console e non System.Console
  - anche in questo caso non ha niente a che vedere con la visibilità delle classi

Namespace

```
namespace Circonferenze {

public class Principale {

   private int SchermoLeggiNumero() {
     System.Console.WriteLine("Quante circ. ?");
     System.Console.Write("---> ");
     int numeroCirconferenze =
            Unibas.Utilita.Console.LeggiIntero();
     while (numeroCirconferenze < 0) {
        System.Console.WriteLine("Errore.");
        System.Console.Write("Ripeti. ---> ");
        numeroCirconferenze =
            Unibas.Utilita.Console.LeggiIntero();
      }
      return numeroCirconferenze;
   }
```

```
namespace Circonferenze {

using Unibas.Utilita;

public class Principale {

   private int SchermoLeggiNumero() {
     System.Console.WriteLine("Quante ?");
     System.Console.Write("---> ");
     int numeroCirconferenze =
            Console.LeggiIntero();
     while (numeroCirconferenze < 0) {
        System.Console.WriteLine("Errore.");
        System.Console.Write("Ripeti. ---> ");
        numeroCirconferenze =
            Console.LeggiIntero();
      }
      return numeroCirconferenze;
   }
```


Namespace

- Attenzione alla differenza
  - import in Java consente di specificare il nome di una o più classi
  - es: import it.unibas.utilita.Console; import java.util.*;
  - using in C# consente di specificare il nome di un namespace e vale automaticamente per tutte le classi del namespace
  - analoga a import usata con l'asterisco

Convenzioni di Stile

- Diverse da quelle di Java
  - ⇨in Java gli unici nomi in maiuscolo sono quelli delle classi
- In C#
  - ⇨i nomi di classi sono in maiuscole
  - ⇨i nomi di namespace sono in maiuscole
  - ⇨i nomi di metodo sono in maiuscole

Convenzioni di Stile

- Esempio: il metodo Main
  - ⇨come in Java l'esecuzione di una applicazione deve cominciare sempre dal metodo Main
  - ⇨ma il nome del metodo deve cominciare con la maiuscola
- Prototipo
  - ⇨public static void Main(string[] args);
  - ⇨public static void Main(System.String[] args);

Convenzioni di Stile

- Conseguenza
  - ⇨mentre in Java è sempre possibile capire cosa viene fatto e da chi nel codice, in C# no
- Esempio
  - ⇨Microsoft.Win32.SystemEvents()
  - ⇨metodo SystemEvents di una classe Win32 del namespace Microsoft (NO)
  - ⇨o costruttore della classe SystemEvents del namespace Microsoft.Win32 (SI)

Convenzioni di Stile

- In generale, però
  - ⇨è opportuno adeguarsi alle convenzioni di stile del linguaggio che si usa
- Quindi
  - ⇨è relativamente semplice tradurre il codice Java in codice C#
  - ⇨ma bisogna sforzarsi di cambiare maiuscole e minuscole per rispettare le convenzioni di stile di C#

La Classe System.Console

- Una classe particolare
  - ⇨la classe Console del namespace System
  - ⇨significativa differenza rispetto a Java
- I due metodi principali
  - ⇨public static void WriteLine (String s); // o Write
    scrive una riga sullo standard output
  - ⇨public static String ReadLine();
    legge una riga dallo standard input

La Classe System.Console

- In altri termini
  - ⇨in Java per lavorare sui flussi standard bisogna usare direttamente i riferimenti System.out e System.in
  - ⇨e per giunta System.in è difficile da usare
  - ⇨in C# è stata fatta una scelta diversa
  - ⇨è stata predisposta una classe apposita per scrivere e leggere dallo standard input

La Classe System.Console

- Però...
  - ⇨la classe System.Console ha dei limiti
- In particolare
  - ⇨consente di effettuare l'input non formattato con il metodo ReadLine
  - ⇨ma non consente di effettuare l'input formattato (ma esiste un metodo ReadInt())
  - ⇨per questo: Unibas.Utilita.Console

```
namespace Calcolatrice {

public class Calcolatrice {

   private double risultato;

   public double GetRisultato() { return this.risultato; }

   public void Somma(double a, double b) { this.risultato = a + b; }

   public void Sottrai(double a, double b) { this.risultato = a - b; }

   public void Moltiplica(double a, double b) { this.risultato = a * b; }

   public void Dividi(double a, double b) { this.risultato = a / b; }

   public void SommaAlRisultato(double b) { this.risultato = this.risultato + b; }

   public void SottraiDalRisultato(double b) { this.risultato = this.risultato - b; }

   public void MoltiplicaPerIlRisultato(double b) { this.risultato = this.risultato * b; }

   public void DividiIlRisultato(double b) { this.risultato = this.risultato / b; }
}

}
```

Classi e Oggetti: C# >> Alcune Differenze

```
namespace Calcolatrice {

public class Principale {

   public void EseguiOperazioni() {
      Calcolatrice calcolatrice = new Calcolatrice();
      bool continua = true;
      while (continua) {
         double a, b, risultato = 0;
         int scelta = SchermoMenu();
         if (scelta == 0) {
            continua = false;
         } else {
            System.Console.Write("   Inserisci il primo argomento:   --> ");
            a = Unibas.Utilita.Console.LeggiDouble();
            System.Console.Write("   Inserisci il secondo argomento: --> ");
            b = Unibas.Utilita.Console.LeggiDouble();
            if (scelta == 1) {
               calcolatrice.Somma(a, b);
               risultato = calcolatrice.GetRisultato();
            }
            ...
            System.Console.WriteLine("\n   Risultato: " + risultato + "\n");
         }
      }
      System.Console.WriteLine("\nArrivederci.");
   }
```


Classi e Oggetti: C# >> Alcune Differenze

```
... // continua
private int SchermoMenu() {
      int scelta;
      System.Console.WriteLine("--------------------------------");
      System.Console.WriteLine("       Calcolatrice             ");
      System.Console.WriteLine("--------------------------------");
      System.Console.WriteLine("   1. Esegui somma");
      System.Console.WriteLine("   2. Esegui differenza");
      System.Console.WriteLine("   3. Esegui moltiplicazione");
      System.Console.WriteLine("   4. Esegui divisione");
      System.Console.WriteLine();
      System.Console.WriteLine("   0. Esci");
      System.Console.WriteLine();
      System.Console.Write("   Scelta ----> ");
      scelta = Unibas.Utilita.Console.LeggiIntero();
      while ((scelta < 0) || (scelta > 4)) {
         System.Console.WriteLine("   Errore. Ripeti ----> ");
         scelta = Unibas.Utilita.Console.LeggiIntero();
      }
      return scelta;
   }

   public static void Main(string[] args) {
      Principale p = new Principale();
      p.EseguiOperazioni();
   }
}
}
```

Gli Esempi

>> calcolatricestatica
>> calcolatrice
>> circonferenze
>> segmenti

- Progetti di riferimento
  - ⇨il codice degli esempi utilizzati in questa parte del corso è molto simile nei due linguaggi
  - ⇨con alcune differenze sintattiche marginali
- Attenzione
  - ⇨cambia invece significativamente l'utilizzo degli strumenti

Il Compilatore di C#

- Uno strumento unico
  - ⇨csc.exe
- Attenzione
  - ⇨a causa della differenza tra namespace e package, il processo di compilazione ed esecuzione è totalmente diverso
  - ⇨in particolare sono diversi i meccanismi per risolvere i riferimenti tra una classe e l'altra

Il Compilatore di C#

- Per ora
  - ⇨vediamo una versione semplificata di questi meccanismi
- La soluzione più semplice
  - ⇨compilare assieme tutte le classi che compongono l'applicazione in modo da collegarle "staticamente"
  - ⇨anche se C# consente il collegamento dinamico come Java

Il Compilatore di C#

- La sintassi
  - ⇨csc <classe1>.cs <classe2>.cs ...
- Esempi
  - ⇨csc Principale.cs Calcolatrice.cs Console.cs
  - ⇨csc Principale.cs Circonferenze.cs ..\Utilita\Console.cs
  - ⇨csc segmenti*.cs utilita\Console.cs

La Documentazione di .NET

- La documentazione di .NET
  - ⇨molto completa
  - ⇨accessibile a partire dal file StartHere.htm in C:\Programmi\Microsoft.NET\SDK\v1.1\
- Attenzione
  - ⇨il file deve essere aperto necessariamente con Internet Explorer
  - ⇨altrimenti i contenuti non sono accessibili

La Documentazione di .NET

>> StartHere.htm

- Struttura dei contenuti
  - ⇨vari file dell'Help di Windows
- Particolarmente interessanti
  - ⇨Introduzione a .NET Framework
  - ⇨Documentazione di .NET Framework SDK
  - ⇨quest'ultimo contiene la documentazione della BCL (Base Class Library) alla voce "Libreria di Classi"

Riassumendo

- Introduzione
- Alcune Differenze
  - ⇨Namespace
  - ⇨Convenzioni di Stile
  - ⇨La Classe Console
- Compilatore di C#
- Gli Esempi
- Documentazione di .NET

Termini della Licenza