Alma Mater Studiorum-Università di Bologna
Scuola di Ingegneria

APPROFONDIMENTO
Le nuove forme di main di Java 25

Corso di Laurea in Ingegneria Informatica

Anno accademico 2025/2026

Prof. ENRICO DENTI

Dipartimento di Informatica – Scienza e Ingegneria (DISI)

IL MAIN IN Java standard

- In Java il main è una funzione *statica, pubblica e void,* avente come argomento obbligatorio un array di stringhe, *collocata dentro una classe* che funge da contenitore

Java

```
public class MyProg {
 public static void main(String[] args){
    int x=3, y=4; int z = x+y;
 }
}
```

Java

```
public class Esempio1 {
 public static void main(String[] args){
    System.out.println("Hello world!");
 }
}
```

IL MAIN IN Java standard

- Questa scelta è chiara e antica, ma verbosa
  - molti linguaggi più moderni la alleggeriscono, rendendo opzionali una o più specifiche
  - Kotlin consente perfino di scrivere il main a top-level, *senza in apparenza scriverlo in una classe (object)*, in quanto il contenitore esterno è definito implicitamente dal compilatore
- Inoltre, costringe a familiarizzare da subito con vari aspetti non fondazionali, danneggiando l'apprendimento
  - chi si avvicina al linguaggio deve conoscere fin dal primo programma i qualificatori **`public`** e, soprattutto, **`static`**
  - per poter scrivere la signature deve inoltre conoscere **`String`** e la notazione ad array **`[ ]`**, *anche se non ha intenzione di usare alcun argomento dalla riga di comando*

IL MAIN IN Java 25

- Per queste ragioni, Java 25 introduce *modi più light* di specificare il main
  - possibilità di omettere i qualificatori **`public`** e **`static`**
  - possibilità di omettere l'argomento **`String[]`**
  - possibilità di omettere anche la classe che lo racchiude, che sarà generata dal compilatore usando *il nome del file*
- In particolare, omettere **`static`** ha conseguenze:
  - significa definire un metodo di istanza anziché una funzione statica
  - provvede il compilatore a creare un'istanza singleton della classe contenitrice (pure generata automaticamente) e invocare su di essa il «nuovo main»

IL MAIN FRA VECCHIO E NUOVO

- Il main classico:

```java
public class Esempio1 {
 public static void main(String[] args){
    System.out.println("Hello world!");
 }
}
```

Java

- La nuova versione più light possibile:

```java
// file ReMain.java
void main(){
    System.out.println("Hello world!");
}
```

Java

  - fra questi due «estremi» sono possibili alcune «vie di mezzo»

IL NUOVO MAIN

- Esperimento

```
C:>javac ReMain.java
C:>java ReMain
Hello world!
```

- Verifica col disassemblatore javap:

```
C:>javap ReMain.class
Compiled from "ReMain.java"
final class ReMain {
  ReMain();
  void main();
}
```

IL NUOVO MAIN: ALGORITMICA

- **Algoritmo di ricerca del «nuovo main»**
  1. il main classico, statico (non necessariamente **`public`**: basta che non sia **`private`**) con classica signature e argomento
  2. il main classico, statico (non necessariamente **`public`**: basta che non sia **`private`**) con classica signature, ma *senza argomento*
  3. un main come *metodo di istanza* (non statico), con argomento
  4. un main come *metodo di istanza* (non statico), *senza argomento*
- Sono quindi accettabili come «nuove forme di main» (con o senza classe che li racchiuda):

```
[public] static void main(String[] args)
[public] static void main()
[public] void main(String[] args)
[public] void main()
```