



Lambda Expressions in Java8: Uso e Tipaggio

Mattia D'Autilia - 5765968 - mattia.dautilia@stud.unifi.it,
Alex Foglia - 6336805 - alex.foglia@stud.unifi.it

05/05/2018

- ❶ Introduzione Java e Java8
- ❷ Lambda Calcolo vs Lambda Java
- ❸ Tipaggio Lambda Java

1.1 : Java

- **Java** è un linguaggio di programmazione di alto livello, principalmente *orientato agli oggetti*, ma accetta anche altri paradigmi come quello *funzionale* ed è a *tipizzazione statica*.
- E' stato creato per soddisfare cinque obiettivi primari:
 - ❶ Essere "semplice e familiare";
 - ❷ Essere "robusto e sicuro";
 - ❸ Essere indipendente dalla piattaforma, da qui il detto "*Write one, run everywhere*";
 - ❹ Contenente strumenti e librerie per il networking;
 - ❺ Essere progettato per eseguire codice da sorgenti remote in modo sicuro.

1.2 : Evoluzione di Java

- Quando Java nacque nel 1995, era un linguaggio molto semplice. Con il passare degli anni sono state introdotte gradualmente tante caratteristiche, diventando un linguaggio sempre più potente e completo, in particolare con la versione 5 e 7. Quello che però non era mai cambiato sino ad ora, era la coerenza d'essere un linguaggio orientato agli oggetti.
- Negli ultimi anni però la scena della programmazione mondiale è cambiata. In particolare con l'avvento di processore multi-core nell'uso domestico, la *programmazione funzionale* è stata rivalutata. Con linguaggi moderni come *Scala* e *Groovy* è possibile scrivere algoritmi con un numero di righe nettamente inferiore, rispetto a quello che si poteva fare con Java, che qualcuno stava già definendo un linguaggio morto in quanto con le versioni 6 e 7 aveva solo modernizzato alcune librerie, estromettendo le tanto richieste *Espressioni Lambda*.

1.3 : Java8

- Con l'avvento di **Java8** fu però apportata una vera e propria rivoluzione, la più innovativa in tutta la storia di Java. Con l'introduzione delle *Espressioni Lambda* e la possibilità di *referenziare i metodi*, la **filosofia funzionale**, fa il suo ingresso nella programmazione Java.
- Ora vedremo come affrontare la nuova sfida, che è quella di far convivere i due paradigmi, quello *orientato agli oggetti* e quello *funzionale*, in modo tale da ottenere il meglio della programmazione.

