Swift

di Anna Maria Gennai

Swift… veloce… voglio scrivere velocemente questo articolo perché sto lavorando a un progetto più complesso e non ho molto tempo. Nemmeno voglio restare indietro con l'abitudine di pubblicare un articolo ogni mese.

Quindi eccomi qui a parlare di Swift, che mi suggerisce l'occasione di scrivere qualcosa per tre mesi consecutivi: Swift per il mese di aprile, Swift per maggio e Swift per giugno.

Più specificamente il linguaggio di programmazione *Swift*, il codice SWIFT e Jonathan Swift.

Swift è stato introdotto da Apple nel 2014 come linguaggio di programmazione per scrivere app con la caratteristica di risultare veloci, da qui il nome. La velocità riguarda soprattutto la rapidità di ricerca dati, così è stato scelto da app quali quelle di Airbnb e di Linkedln.

Oltre ad essere un software open source, quindi di libera accessibilità, il codice di programmazione *Swift* é anche facile da imparare e da applicare. Per questo è stato utilizzato come linguaggio in numerosi corsi di avviamento alla programmazione diffusi in tutto il mondo.

Come descritto dalla stessa Apple, si tratta di una risorsa semplice ed intuitiva, "per facilitare lo sviluppo di app per iOS, Mac, Apple TV e Apple Watch, e progettato per offrire agli sviluppatori una libertà mai vista prima".

Subito dopo il suo lancio, la curiosità dei programmatori nello sperimentare il nuovo prodotto spinse *Swift* ai primi posti della classifica dei linguaggi più utilizzati per le piattaforme Apple. Col passare degli anni è stato superato, o risuperato, da altri. Secondo le indagini condotte in tutto il mondo da Linkedln Job, i primi otto programmi più richiesti nel 2023 vedono, nell'ordine, Pyton, Java (davanti a Pyton nella classifica europea), JavaScript, C++, C#, C, TypeScript, PHP. Tuttavia *Swift* risulta importante per chi intenda realizzare app per iPad, iPhone, Mac e per tutti i dispositivi con sistema operativo Linux.

Oggi esistono numerosi linguaggi di programmazione, che permettono di interpretare i nostri comandi su computer, tablet, cellulari e realizzare siti web o app, e tradurre i nostri intenti in istruzioni comprensibili dai dispositivi digitali. Ciascuno ha caratteristiche che lo rendono più performante in un ambito piuttosto che in un altro, ad esempio a seconda del sistema operativo, Android, iOS, Linux. Oltre il 90% dei siti web è realizzato in *JavaScript*, invece *Python* è il più utilizzato per il trattamento di Big Data e intelligenza artificiale, *Java* per l'*IoT* (*Internet of Things*, applicazioni che attraverso la connessione a grandi database permettono di gestire e controllare dispositivi di uso quotidiano). Nonostante alcune differenze, tutti i linguaggi hanno una sorta di "scheletro" comune: i programmi, comunque scritti, devono avere un "inizio" e una "fine" e contenere una successione finita di istruzioni che possono essere di assegnazione, di ciclo, di selezione, etc.. per ottenere il risultato desiderato. Il primo programma della storia fu scritto alla metà del XIX secolo da Ada Lovelace, la

figlia di Lord Byron (si può approfondire l'argomento su *studiomatematica* all'indirizzo http://www.studiomatematica.it/doc/byron%20ultimo.pdf), ma l'escalation della comunicazione con gli elaboratori si ebbe a partire dagli anni sessanta. Il primo linguaggio evoluto, ovvero strutturato in modo da poter poi essere a sua volta tradotto dalla macchina, per mezzo di un altro programma, in una successione di cifre 0 e 1 che costituiscono il codice con cui sono memorizzate nel computer tutte le istruzioni, fu il *FORTRAN* (da FORmula TRANslator) grazie al quale venivano scritti programmi scientifici. Si impostava matematicamente o fisicamente un problema complicato dal punto di vista computazionale, poi si traduceva in Fortran, lo si scriveva al computer e si mandava il programma in esecuzione, in modo che la macchina eseguisse quei calcoli che sarebbero stati estremamente complessi da eseguire manualmente. Per la stesura del codice a volte occorrevano giorni e giorni, perché era necessario testare l'attendibilità del programma su esempi risolvibili manualmente, per essere sicuri che i risultati sarebbero stati quelli attesi anche in situazioni improponibili anche per la persona più capace nei calcoli. Quando il programma non dava gli effetti previsti, andava ricercato l'errore analizzando una istruzione per volta, cercando di capire perché e dove l'algoritmo si inceppava, entrava in loop, oppure forniva un valore inesatto. La versione che utilizzavo quando lavoravo al CNUCE (Centro Nazionale Universitario di Calcolo Elettronico) di Pisa era *Fortran IV*. Un problema complicato, come ad esempio quello necessario alla messa in orbita di un satellite, può essere scomposto in una serie di sottoproblemi, traducibili in sottoprogrammi, o *subroutine*, esaminabili singolarmente.

Tutti i programmi scritti con i cosiddetti linguaggi evoluti hanno caratteristiche affini, per cui gli operatori in grado di tradurre un problema in uno di essi non hanno difficoltà a farlo anche con gli altri.

Approfondiremo l'argomento in un prossimo articolo.