Guida alla programmazione (volume 1)

evilsocket@gmail.com
http://www.evilsocket.net/

Salve a tutti gente !
Oggi affronterò un argomento che quasi nessuno affronta di solito ovvero ... cosa è la programmazione ?
Quanti programmatori che scrivono codice senza sapere cosa stanno facendo esistono ? TROPPI ! :D
Quindi, in questo articolo particolarmente teorico, andremo ad affrontare delle tematiche molto molto
basilari, ma che paradossalmente mancano nella mente di molti presunti programmatori ^^ .
Io direi, prima di tutto, di iniziare con la definizione più semplice che mi viene in mente se penso a cosa
è un programma ...
Un programma è un insieme di dati ed istruzioni che hanno lo scopo di assolvere uno o più compiti
Niente di troppo astruso no ? ^^
Quando per la prima volta misi le mani su un pc e mi venne in mente questa definizione, dissi tra me e me : "Be,
se provo ad aprire un programma con un editor di testo li vedrò sti dati e ste istruzioni !!!" (nn prendetemi x il culo,
ero solo un bimbo XD)
Ora, se proviamo ad aprire un programma con un editor di testo, vedremo una marea di caratteri strani ... Tranquilli !
Il vostro computer non usa istruzioni in aramaico !!! :D
Quello che vedete davanti a voi, sono le famose istruzioni, però codificate in un modo tale che il processore possa
capirle (questa cosa la approfondiremo più avanti ovviamente).
A questo punto uno si potrebbe chiedere : "ECCHECAZZO ! Ma per fare un programma bisogna mettersi a scrivere tutti
quei caratteri strani ?!?!?!" Beh, mi sembra evidente che non è così, vabbè che noi programmatori siamo strani, ma cazzo
mica fino a questo punto :D .
Proprio per ovviare a questo problema, ovvero la difficoltà di dialogo tra l'uomo ed il processore, sono nati i cosiddetti
LINGUAGGI DI PROGRAMMAZIONE .
Mettiamola così, il nostro cervello funziona emettendo, ricevendo ed interpretando impulsi elletrochimici
giusto ?
Quando un vostro amico vuole dirvi qualcosa, non è che si mette con un cacciavite nella vostra testa a smanettare con i vostri neuroni :D ma per l'appunto usa un LINGUAGGIO a voi comprensibile che il vostro cervello INTERPRETERA' trasformandolo
nei suddetti impulsi .
Ebbene, la stessa cosa vale per i linguaggi di programmazione ... non sono altro che un TRAMITE tra il mondo esterno e la cpu .
Un modo che i programmatori hanno per "parlare" con la cpu in un linguaggio comprensibile ad entrambi .
Seguendo la scia del mio stesso esempio, vi faccio notare un' altra cosa ...
Come l'italiano o l'inglese hanno una sintassi (essì cavolo,

quanto mi rompeva le palle la
prof sulla sintassi dei miei
temi XD), anche i linguaggi di programmazione hanno una loro
sintassi, e se questa sintassi non viene rispettata, sono cazzi
acidi !
(tanto per citare Baby_Vegeta :D).
Qui entra in gioco un altro elemento della nostra storia, il
COMPILATORE ... chi sarà mai questo sconosciuto !?!?!?
Ebbene, una volta scritto un programma in un determinato
linguaggio dentro un semplice file di testo, per quanto perfetta
la nostra
sintassi possa essere, la cpu non è ancora in grado di capirlo ...
quindi ci servirà qualcosa che "trasformi" questo nostro
file di testo (chiamato codice sorgente) in un file eseguibile
comprensibile alla cpu .
Questo "qualcosa" si chiama, appunto, COMPILATORE .
Questa entità (lui stesso è un programma) trasformerà il nostro
codice sorgente in quell insieme di caratteri a noi
incomprensibili
ma che piacciono tanto al nostro processore :D .
Avete presente la mia prof che si incazzava se facevo un errore di
sintassi nei temi ? Beh, il compilatore non è da meno ragazzi !
Immaginatelo come un professore ... se il sorgente che gli date da
compilare (diciamo il tema ^^) ha una sintassi sbagliata,
solitamente esce fuori un messaggio del tipo "We testina ! Ma che
cazzo hai scritto qua ? E' meglio se impari ad usare la
caffettiera
prima di scrivere sta specie di codice !" ... vabbè, non è che
dice ESATTAMENTE questo, ma avete capito il senso no ? Come no ?
Non avete capito questo geniale paragone ?!?!?!? Insomma, il
compilatore vi segnala che in una determinata riga del file che
gli avete
dato in pasto c'è un errore di sintassi .... madò quanto siete
poco perspicaci però eh !!! :D
Dopo aver corretto eventuali errori di sintassi, riproviamo a dare
in pasto il sorgente al compilatore e ... voilà ! Abbiamo il
nostro
bel file eseguibile :D
C'è da fare ora una piccola (mica tanto ^^) precisazione ... in
realtà, non è così per TUTTI i linguaggi, ci sono due principali
categorie di linguaggi :
*.: I linguaggi compilabili (come il C/C++, il Visual Basic, il
Delphi, etc)*
*.: I linguaggi interpretati o di scripting (come il PHP, il Perl,
il Python, etc)*
Nella precedente sezione di questo articolo, come avrete
immaginato, abbiamo preso in esame i linguaggi della prima
categoria, nella quale partendo da un codice sorgente ci
ritroviamo con un file eseguibile .
Nella seconda categoria invece, abbiamo sempre dei file sorgenti,
ma non abbiamo nessun eseguibile ... in questo caso, un
INTERPRETE prende il posto del compilatore, ed invece di
trasformare il sorgente in un eseguibile, lo interpreta ed esegue

lui stesso
le istruzioni presenti nel file .
Per quale motivo questo ? Beh, è una problematica che si è discussa per decine di anni quando si era agli albori dell informatica .
Se io produco un file eseguibile sotto Windows, non potrò usarlo sotto Linux per esempio, perchè il compilatore creerà del codice comprensibile solo sotto i sistemi di zio Bill ... ora, questo non sempre va bene, spesso si vuole scrivere codice che sia PORTABILE su ogni sistema operativo ... e qui vengono in nostro aiuto i linguaggi di scripting, quelli della seconda categoria .
Se su ogni sistema operativo è presente l'interprete di un dato linguaggio di scripting, noi possiamo scrivere il nostro programma sotto Windowz, per poi poterlo eseguire sotto Linux perchè tanto abbiamo su entrambi i sistemi un interprete che si occupa di gestire
le differenze tra i due sistemi operativi ... comodo no ?
Sicuramente si, ma dove un programma scritto con un linguaggio di scripting è utilizzabile su diversi sistemi operativi, dovendo ogni volta essere interpretato da un altro programma è nettamente più lento di un linguaggio compilato .
Questo perchè, un programma trasformato in eseguibile da un compilatore, viene direttamente eseguito dalla cpu, senza altri programmi
che fanno da tramite, l'unico tramite ci sarà durante la compilazione iniziale, ma fatta quella il nostro eseguibile potrà girare
senza ulteriori passi .
In realtà, ci sarebbero anche dei linguaggi "ibridi", ovvero una via di mezzo tra la prima e la seconda categoria, come Java o i linguaggi del framework .NET, ma essendo dei casi molto molto particolari, è meglio ignorarli per adesso :D .
Ok ragazzi, per questa prima parte è tutto, nella prossim vedremo i concetti di
"variabile" e della gestione della memoria.
Come al solito spero che il mio articolo sia stato, seppur tratti un argomento molto complicato, il più chiaro possibile .
Considerate che lo sto scrivendo dopo una notte insonne ed un weekend di alchool, sesso e droga sulle spalle, quindi non siate troppo
duri nei giudizi ^^ .
Se avete bisogno di delucidazioni o semplicemente volete mandarmi a quel paese, sentitevi liberi di postare su questo forum e sarà mio
piacere rispondere a quesiti o insulti di sorta :D .
Mi raccomando, continuiamo a condividere il sapere, perchè è un bene che appartiene a tutti quanti .

evilsocket