Il Resto del Pinguino - N° 1 - Maggio 2010 - Pubblicazione aperiodica della comunità italiana dei programmatori Gambas - www.gambas-it.org

Il Resto Del Pinguino

Open Source, Free Software e Programmazione

1

Ubuntu Lucid Lynx 10.04: finalmente è arrivata!

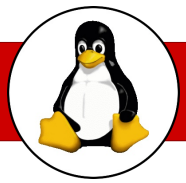
MYPAINT
Risveglia l'artista che c'è in te

GUIDA AL C
Corso di programmazione in C

GAMBAS
Aggiungere i menu programma alle applicazioni

CC BY NC SA

Editoriale



Nel momento in cui sto scrivendo i download del n° 0 de "*Il resto del Pinguino*" hanno superato quota 600! É un traguardo veramente notevole, considerando che l'e-zine è nata da pochissimo, che Gambas è un prodotto di nicchia e che il seguito di programmatori che ha non è certo ai livelli di quelli del Python, tanto per fare un nome. Quindi il passaparola è stato veramente notevole visti i risultati ottenuti. Quindi un "grazie" particolamente caloroso va senz'altro a tutti coloro che hanno scaricato la nostra pubblicazione ma anche a tutti coloro che hanno "diffuso il verbo".

Ed ora veniamo all'articolo di copertina, che riguarda **Ubuntu 10.04**, nome in codice "*Lucid Lynx*". Volenti o nolenti, Canonical si è imposta negli ultimi anni sul mercato con un sistema operativo veramente notevole, capace di strappare utenti agli altri sistemi ed avvicinare tante persone al mondo opensource. L'ultima fatica dei ragazzi di Mark Shuttleworth è un prodotto veramente interessante, a cominciare dalla rinnovata veste grafica e da tante piccole semplificazioni che il nostro Pixel ha analizzato per voi nel suo articolo di pag. 5. Un prodotto, forse, finalmente capace di imporsi anche a chi ricerca le comodità di Windows ed ha timore a passare al Pinguino perché non vuole "sporcarsi le mani" (leggi: intervenire anche in minima parte tramite terminale per configurare il sistema). Senza dimenticarsi che questo è un rilascio **LTS**, vale a dire con supporto per la versione desktop di ben 3 anni.

Una piccola nota anche sul'**iPad**. Anche stavolta Steve Jobs ha colto nel segno ed ha scosso il mercato con uno di quei prodotti rivoluzionari di cui solo lui è capace. E' forse, questo, l'inizio della rivoluzione che cambierà nei prossimi anni il nostro modo di intendere i computer? Chi lo sa, tutto dipende da come sarà accolto l'iPad. Per ora, comunque, le aspettative sono ottime: si parla già di centinaia di migliaia di prenotazioni d'ordine già consegnate. E che la strada sia probabilmente quella giusta lo dimostra il fatto che anche Google seguirà Apple presentando a breve un prodotto similare, che qualcuno già scherzosamente ha ribattezzato **gPad**.

Accenniamo anche al fronte Gambas: questo linguaggio è ormai maturo e capace di grandi imprese. La conferma viene da *Guygle*, una piattaforma web 2.0 con la quale l'amministrazione di Parigi sta catalogando tutte le decine di migliaia di segnali stradali della città. L'applicativo lato server è infatti scritto in Gambas da Benoît Minisini, il "papà" di Gambas (`http://gambas.sourceforge.net/`).

Ed ora.... buona lettura!

Leonardo "leo72" Miliani

REDAZIONE

Impaginazione: Leonardo "Leo72" Miliani - `leonardo@leonardomiliani.com`
Coordinazione articolisti: Francesco "Ceskho OpenCode" Apruzzese - `cescoap@gmail.com`
Grafica: Fabio "Pixel" Colinelli - `pixel.ubuntu@gmail.com`

COLLABORAZIONI

Se volete pubblicare un articolo sulla rivista, inviate la vostra opera (preferibilmente in forma di pacchetto compresso) all'indirizzo **`ilrestodelpinguino@gambas-it.org`** in formato testuale o, in alternativa, in formato ODT (OpenOffice), con eventuali foto come allegato. Nota: l'invio del materiale non obbliga la redazione alla sua pubblicazione. Sarà nostro insindacabile giudizio pubblicare o meno l'articolo ricevuto con eventuali tagli e modifiche dettati da motivi tipografici. Con l'invio del materiale l'autore dà tacito consenso all'utilizzo della sua opera ai fini della pubblicazione. Si ricorda che saranno pubblicati solo gli articoli inerenti i temi trattati dalla rivista.

Sommario

Il Resto del Pinguino

Num. 1 - Maggio 2010

4. News
Notizie e curiosità dal mondo dell'informatica

5. Ubuntu Lucid Lynx 10.04 LTS
Uno sguardo alla nuova versione della nota distribuzione di Canonical

9. Guida alla programmazione in C
Corso a puntate di di programmazione in linguaggio C

12. Su che cosa e come realizzare un programma *(2ª parte)*
Il computer come intelligente strumento di sussidio alle attività umane

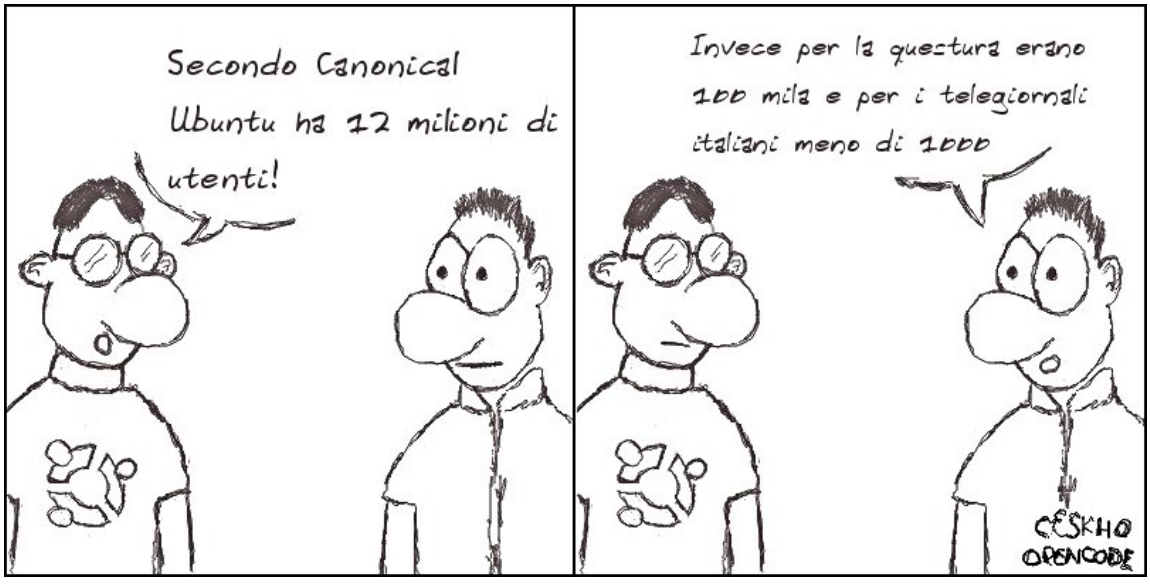
14. MyPaint
Risveglia l'artista che c'è in te

16. I menu con Gambas
Aggiungere i menu programma alle vostre applicazioni

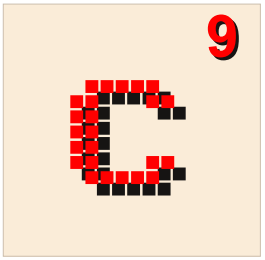
17. La funzione Replace()
Come sostituire parti di testo all'interno delle stringhe

19. Recensioni
I libri consigliati da "Il resto del Pinguino"

20. La cattedrale ed il bazar *(2ª parte)*
Due stili di sviluppo del software a confronto: il modello "cattedrale" e quello "bazar"



Ubuntu 10.04 Lucid Lynx



Guida al linguaggio C

MyPaint

La funzione Replace()

News

Novità e curiosità dal mondo informatico

RILASCIO 1.8 PER WESNOTH

Nonostante si evidenzi sempre il fatto che GNU/Linux pecca per troppe mancanze nel settore videoludico, bisogna anche dire che quei pochi giochi che ci sono sono bene presenti e son pure fatti bene. È il caso di Wesnoth, che da anni intrattiene i videogiocatori pinguineschi di tutto il mondo, che è attivissimo sotto il punto di vista dello sviluppo. È stata infatti rilasciata la versione 1.8. La nuova versione prevede tante novità: è stata introdotta la campagna "Le Memorie di Delfador", alcune storie sono state riviste e migliorate, nuove basi musicali si aggiungono a quelle già esistenti.

Se siete interessati potete visitare il sito ufficiale: http://www.wesnoth.org/

FINALMENTE OPENTTD 1.0.0

Dopo 6 anni di sviluppo vede finalmente la luce la versione 1.0.0 del celebre gioco openTTD che si pone l'obiettivo di essere la controparte open del gioco Microprose Transport Tycoon Deluxe. Tante novità giungono nella versione a cifra tonda. Mappe più grandi, intelligenza migliorata e server dedicato sono solo alcune delle nuove funzioni che è possibile trovare.

Per maggiori info: http://www.openttd.org/en/

IL CUORE DEL SISTEMA SI AGGIORNA

Chi non conosce gcc? Le tre lettere magiche sono sicuramente tra le più digitate nei terminali dei linuxiani che amano programmare. La relase candidate è la 4.5 che vedrà la luce in ritardo rispetto al periodo prefissato. Tale versione prevede il supporto al MPC (MultiProcesso), la compilazione per i nuovi processori ARM, supporto al nuovo standard C++0x ed altro ancora. Potremo gustarci il compilatore sotto il solleone estivo, quando verrà rilasciata la nuova versione, e sicuramente nelle nuova versione delle nostre distro preferite.

Per maggiori info sulle nuove funzionalità è possibile consultare il sito: http://gcc.gnu.org/gcc-4.5/changes.html

APPLE iPAD

Alcune settimane fa Apple ha presentato la sua nuova creatura tecnologica, l'**iPad**. Si potrebbe semplificare pensando all'iPad come ad un iPod di grandi dimensioni anche se le capacità del nuovo prodotto sono ben superiori. Rispetto all'iPod ciò che balza subito all'occhio sono le dimensioni ben maggiori (circa quelle di un foglio A4) grazie alle quali l'utente può navigare, sfogliare foto e lanciare applicativi (gli stessi per iPod e iPhone) con un coinvolgimento ed una esperienza visiva superiori. In vendita in America a fine aprile ad un prezzo di circa 499$.

GOOGLE gPAD

Girano indiscrezioni secondo le quali l'iPad non rimarrebbe solo sugli scaffali per molto tempo. Secondo voci accreditate la società di Mountain View ha deciso di presentare un prodotto similare che è stato ribattezzato dai media **gPad**. Il prodotto di Apple, nonostante il buon numero di prenotazioni, presenta alcuni lati negativi: non ha webcam, porte USB ne' slot per memorie SD, può far girare solo gli applicativi acquistati sull'AppleStore e non è capace di riprodurre filmati Flash. Il gPad dovrebbe offrire ciò che manca all'iPad a livello di hardware e, grazie al fatto che è basato su un sistema aperto (Android), non avrebbe teoricamente limiti di espandibilità software.

Ubuntu Lucid Lynx 10.04 LTS

Le novità della nuova versione della nota distribuzione di Canonical

È uscita **Ubuntu Lucid Lynx 10.04 LTS**.

Con la frase sopra riportata si potrebbe ritenere conclusa la recensione di questa nuova versione di una delle distribuzioni più utilizzate in ambito desktop.

Se state leggendo queste parole avete in mano il numero 1 de "***Il Resto del Pinguino***" e si presuppone che conosciate già un sistema operativo Linux con annessi pregi e difetti; nonostante questo vedrò di illustrare di seguito i cambiamenti apportati da Ubuntu in questa nuova realizzazione:

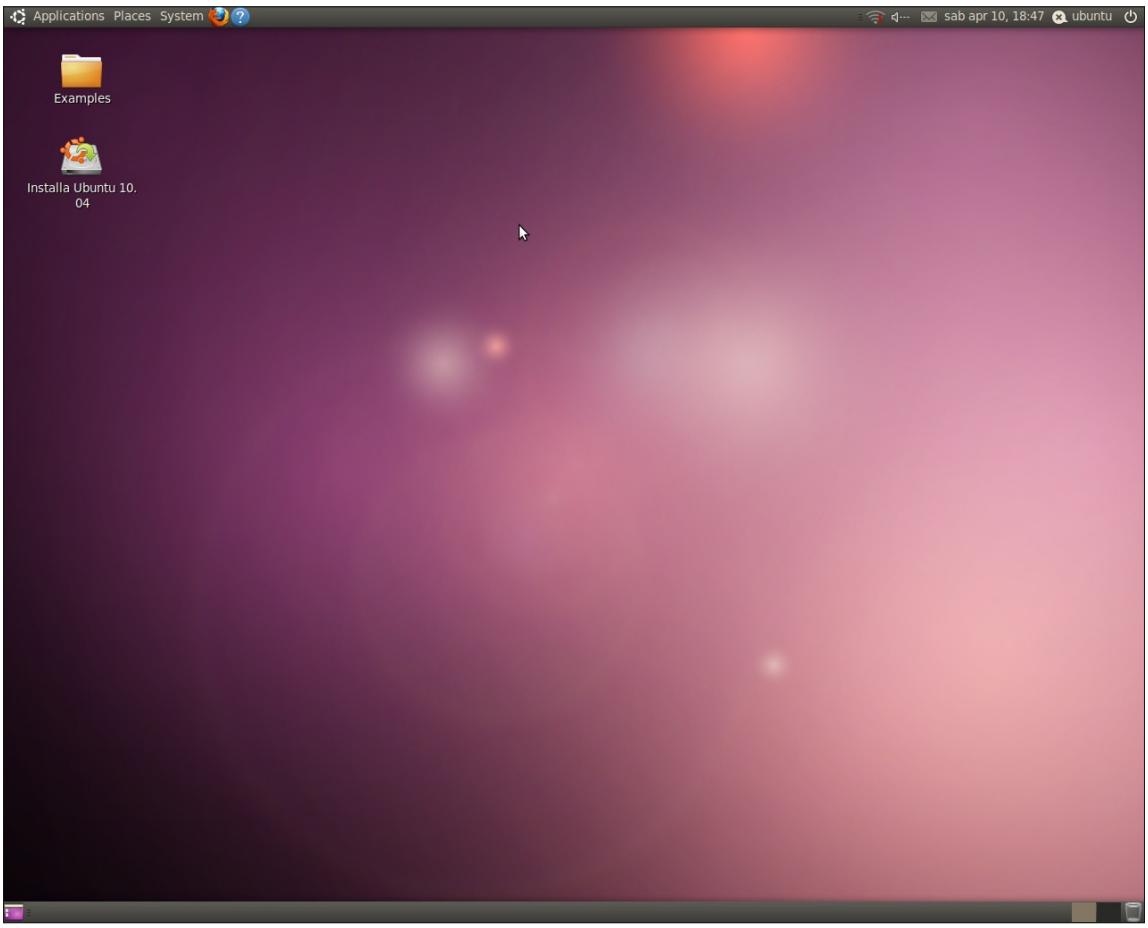
Kernel Linux 2.6.32, Gnome 2.30 .....

No, non ci siamo, non me ne può fregar di meno di queste sigle astruse che semplicemente ci dicono quale versione di questo o quell'altro software è installato sul nostro sistema, se sono un utente esperto probabilmente avrò già deciso quale distribuzione soddisfa maggiormente le mie esigenze, se sono un utente con sufficienti conoscenze mi basterà sapere quali strumenti o semplificazioni sono state apportate per rendere più agevole l'utilizzo del PC e se sono un utente che ha sempre usato Windows e non sa nulla di GNU/Linux a maggior ragione parlare di Kernel, Gnome o altro avrà la stessa valenza di parlare di marmellata ad un convegno sull'energia nucleare.

Quindi vediamo in soldoni cosa succederà se avrò il coraggio di inserire il CD di Ubuntu all'interno del mio PC.

All'avvio potrete scegliere quale lingua utilizzare (consiglio l'italiano, evitate l'aramaico antico), dopo alcuni minuti di caricamento del sistema vi troverete innanzi a questa schermata:

LTS, o ***Long Time Support***, è la sigla con cui Canonical identifica i rilasci del suo sistema operativo con un supporto a lungo termine: 3 anni per le versioni desktop e 5 per quelle server.



La nuova veste grafica dell'ultima versione di Ubuntu

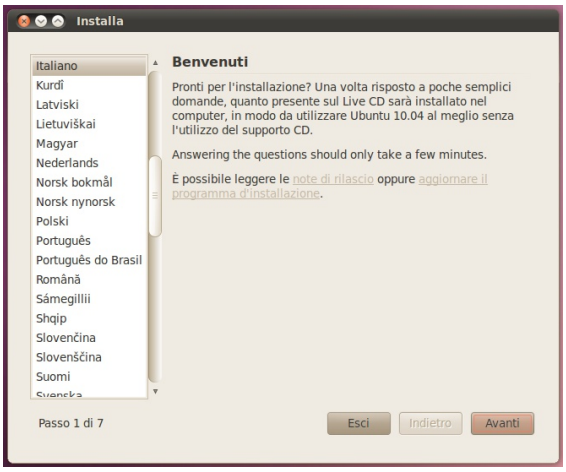
Le opzioni sono tre:

A) ma che è sta roba?

B) Carino, facciamo un giro di perlustrazione.

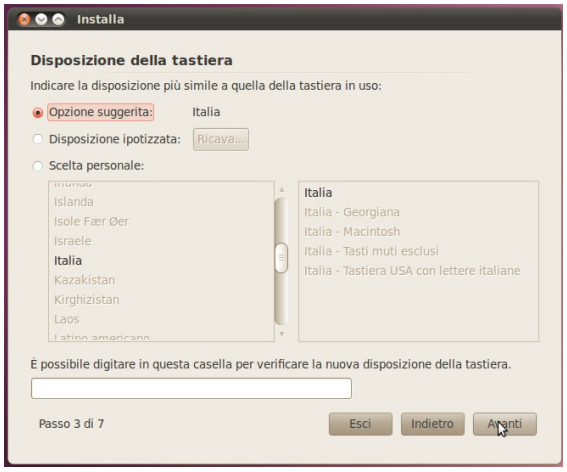
C) Installa Ubuntu 10.04.

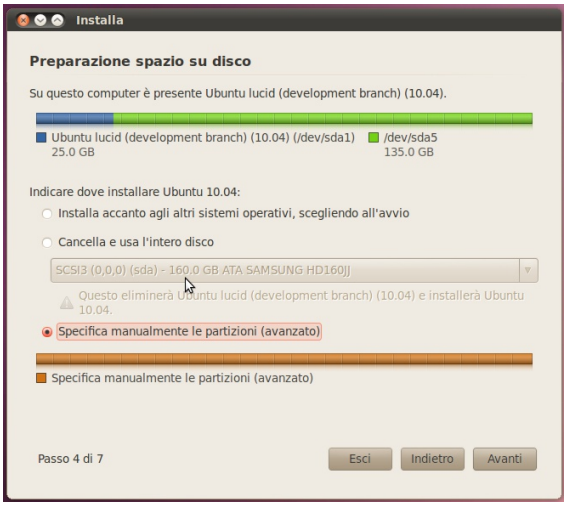
Scartando la "A" per esigenze di redazione e la "B" perché non sarebbe attendibile in termini di prestazioni, procediamo decisi verso la "C".



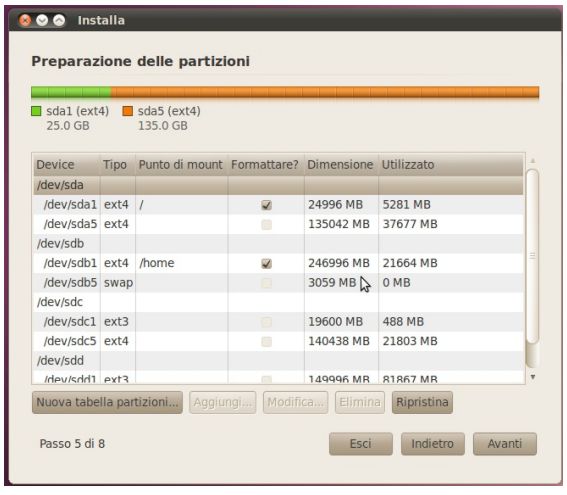
L'installer sviluppato da Canonical è da sempre uno dei più semplici e chiari. Con pochi click ed in meno di 30 minuti anche un utente alle prime armi riesce a configurare ed installare il suo nuovo sistema Ubuntu, anche a fianco di un altro sistema operativo.

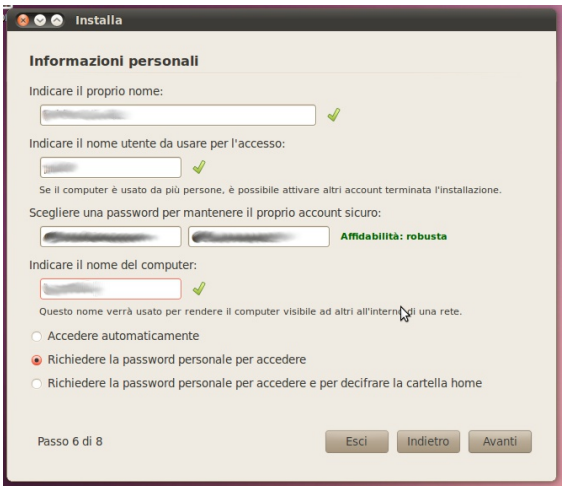
L'installazione di Ubuntu avviene in maniera semplice e rapida. Si sceglie la propria località, la lingua del sistema...





...la disposizione della tastiera, come partizionare i dischi (se in modo automatico...





...oppure in manuale) e poi alla fine si inseriscono i dati del proprio account.

Come si può notare il sistema di installazione è rimasto invariato dalle precedenti versioni, le opzioni di scelta non lasciano dubbi e permettono un sufficiente grado di personalizzazione.

ATTENZIONE: prima di procedere si consiglia sempre di eseguire una copia di sicurezza dei propri dati (foto di matrimonio, documenti, filmati ecc...). Per quanto la procedura sia pressoché sicura può sempre accadere l'evento imprevisto ed è buona norma essere preparati in tal caso.

Dopo aver deciso come installare la nostra Ubuntu ed aver inserito le informazioni personali (nome, utente e password) possiamo procedere e prenderci un caffè nell'attesa che il sistema venga installato sul PC.

Se invece siete di quelle persone che non vogliono perdersi nemmeno un secondo dell'installazione e che sono convinte che guardando intensamente il monitor il PC sia più veloce, vi gusterete questa carrellata di informazioni ed immagini che cercheranno di allietare e rendere più piacevole l'attesa:

Al termine dell'installazione vi sarà chiesto se volete continuare ad usare il sistema da CD oppure riavviare ed usare quello appena installato.... che domanda... ovviamente riavviamo!!!

L'avvio è veloce ed in pochi secondi vi trovate alla finestra di login dove, dopo aver selezionato il vostro nome, inserite la password ed accedete.

Vediamo ora che cosa hanno combinato dentro questa distribuzione:

1) semplificazione scansione immagini

2) rimozione di GIMP

3) software di montaggio video

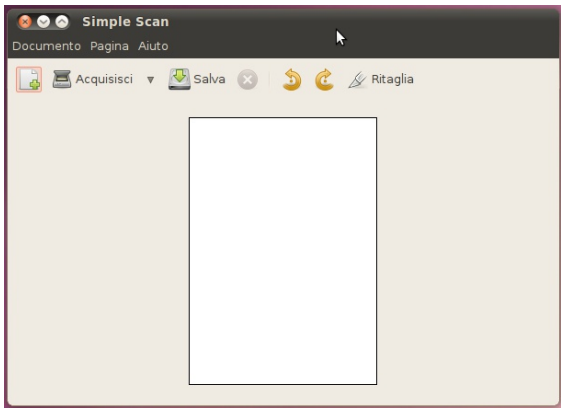
4) semplificazione account di messaggistica

5) nuovo tema grafico

6) Ubuntu Software Center migliorato

1) semplificazione scansione immagini

Beh.. più semplice di così...

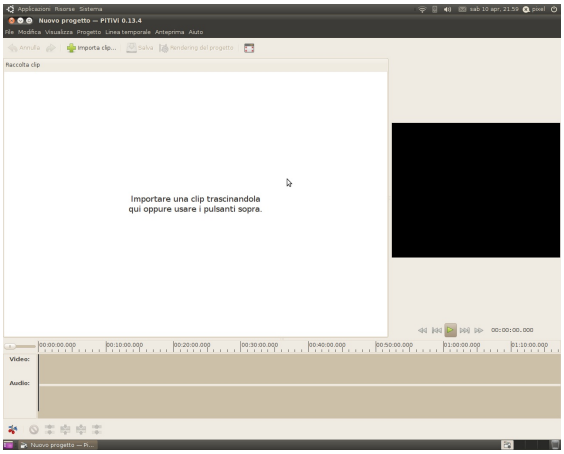


2) rimozione di GIMP

Non c'è nulla da dire a riguardo, se lo cercate nel menù grafica non lo troverete, se non potete farne a meno lo potrete installare mediante il Software Center.

3) software di montaggio video

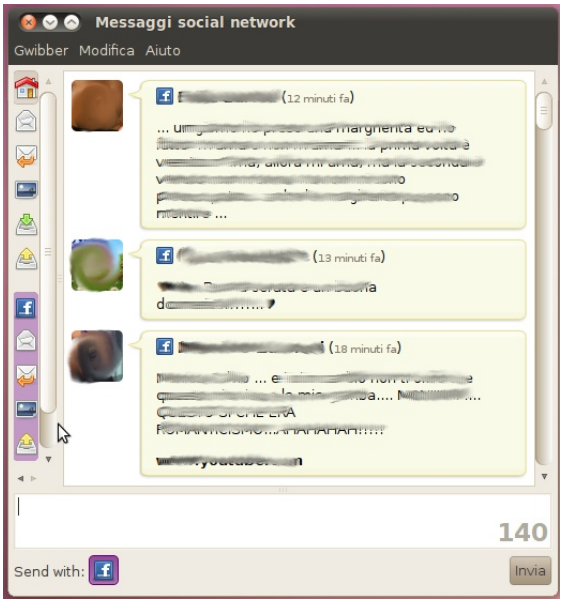
Anche qui non si scherza in termini di semplicità. È chiaro che non vincerete un Oscar per il miglior montaggio ma sicuramente il filmino di Natale potrà essere un po' più arricchito (senza esagerare).

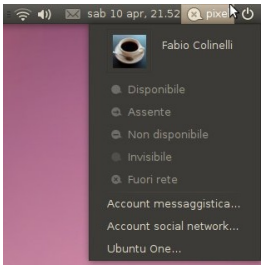


Le novità di questa ultima versione di Ubuntu sono molte, e tutte importanti. La più appariscente resta comunque il cambio del tema grafico, qualcosa che gli utenti chiedevano a Canonical a gran voce da molto tempo.

4) semplificazione account di messaggistica

Non c'è nulla da fare, tutti ci vogliono sempre online ed Ubuntu non è da meno. Selezionando il nostro nome utente ci verrà mostrato un simpatico menù dove poter aggiungere vari e variegati account da Facebook a MSN passando da Twitter e terminando a Identi.ca.





Una sola icona nella sys-tray per gestire i propri account di messaggistica istantanea e di social networking.

5) nuovo tema grafico

Non ditemi che non ve ne eravate ancora accorti. Ebbene sì, Ubuntu ha deciso di rifarsi il *look* e devo ammettere che quello nuovo è riuscito decisamente bene. Ovviamente ognuno ha i propri gusti estetici e sicuramente i miei cozzeranno contro i vostri ma di una cosa posso essere certo: la strada intrapresa denota un forte studio a monte e nulla viene lasciato al caso (quasi nulla).

6) Ubuntu Software Center migliorato

Non ho mai amato queste cose semplicistiche ed ho sempre preferito il tradizionale Synaptic (se non addirittura il terminale) ma devo ammettere che funziona decisamente bene:

Conclusioni e pareri

Ubuntu Lucid Lynx 10.04 è una distribuzione di primo impatto ben curata sotto l'aspetto grafico ma posso assicurarvi che molto è stato fatto anche per renderla affidabile e sicura. La voglia di semplicità trasuda da tutti i bit e l'utente viene letteralmente accompagnato mano nella mano fino al raggiungimento di tutte le fasi (dall'installazione alla configurazione). Io l'ho installata sia su un desktop P4 3 GHz HT, ATI X1650, 2 GB RAM che su un Acer Aspire 5735Z e non ho riscontrato nessun problema. Reattiva e precisa, sembra che tutto sia sempre sotto controllo.

Come sempre i software presenti sono aggiornatissimi ed essendo la 10.04 una **LTS** saranno garantiti aggiornamenti di sicurezza e critici per ben 3 anni.

Nei mesi passati ho seguito quasi giornalmente l'avanzamento dei lavori e mai come questa volta ho notato la voglia di perfezione, nulla è stato lasciato al caso. All'interno della comunità italiana si è formato un gruppo specifico "ubuntu-it test" capitanato dal sempre presente *Paolo "xdatap1" Sammicheli* che ha svolto verifiche continue e mirate con il compito di segnalare bug ed anomalie sulle versioni live.

PRO

Gratuita

Aggiornata

Esteticamente gradevole

Supportata ottimamente

Facile

Operativa immediatamente

Ottimo riconoscimento hardware

Supporto e sincronizzazione con IPhone (servizio a pagamento)

Ubuntu One per acquistare musica in rete (servizio a pagamento)

CONTRO

Il caffè ancora non riesce a farlo

Fabio "Pixel" Colinelli

Guida al C in puntate - Parte 1a

Impariamo a programmare in C, linguaggio sempre attuale

La programmazione è sicuramente una delle possibilità più affascinanti che l'informatica può offrirci. Essa ci permette di essere come un genitore che, mediante le proprie parole e le proprie indicazioni, insegna al suo bambino come deve comportarsi con le sollecitazioni che giungono dal mondo esterno. Saper programmare non vuol dire solo conoscere le parole chiave che ci permettono di gestire i nuovi elementi della tecnologia ma vuol dire dare al comportamento umano e naturale una struttura interpretativa algoritmica.

Chiunque inizia a programmare lo fa per uno scopo preciso che può variare dalla conoscenza personale alla passione, dal lavoro all'utilità. Chi si avvicina a questo mondo lo fa attraverso il linguaggio che ritiene più idoneo per le proprie esigenza e fortunatamente il mondo informatico pulula di scelte tra le quali poter vagliare quella migliore. Esiste però un linguaggio che nel corso della storia informatica ha saputo ritagliarsi il posto di padre di tutti i linguaggi, ovviamente sto parlando del C.

Il C è il linguaggio più snobbato dai neofiti ma diventa indispensabile nel momento in cui lo si impara a conoscere e ad usare. Esso gode di particolari costrutti e particolari elementi che lo rendono adatto a tutti i tipi di obiettivi software.

Il fine del nostro percorso sarà quello di imparare le basi di questo linguaggio in modo che il programmatore, che si ritrova a consultare queste pagine, sappia dare forma alle propria idee grazie alla propria macchina ed a questo linguaggio.

Perché scegliere il C

I motivi per cui bisogna scegliere il C sono molteplici e tutti egualmente importanti.

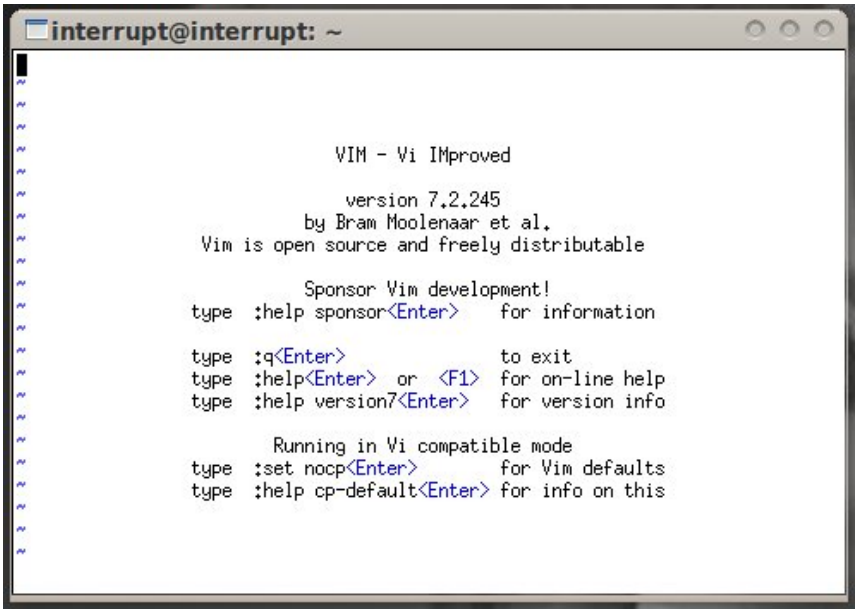
Sicuramente un punto a suo favore è il fatto che esso sia di tipo compilato. Un linguaggio compilato è un particolare linguaggio di programmazione il cui codice viene passato ad un software detto compilatore che provvede a trasformarlo in linguaggio macchina. In questo modo si avrà un codice che comunica quasi direttamente con l'hardware su cui gira rendendo il proprio programma finale più performante è meno avido di risorse poichè costruito ad hoc per la macchina su cui si vuole che esso funzioni.

Una caratteristica di rilevante importanza del C sono gli strumenti che esso propone. Grazie a dati come i puntatori, ad esempio, sarà possibile manipolare le variabili richiamandole direttamente dall'indirizzo preciso in cui risiedono in memoria. Sarà possibile creare elementi che ricalchino la propria idea di dato ridefinendo quelli già esistenti o sarà possibile procurarsi della memoria per le proprie variabili anche lì dove apparentemente pare impossibile.

Infine non bisogna dimenticare che il C è multi-piattaforma; questo vuol dire che il codice scritto su un computer con sistema operativo GNU/Linux può essere eseguito senza problemi e senza alcuna modifica su sistemi Microsoft, Mac, Solaris e via discorrendo. Ovviamente, però, bisogna tener presente che se nel proprio codice si implementano delle chiamate dirette a dei comandi di sistema tale proprietà viene a decadere.

Cosa serve per programmare in C

Programmare in C è, oltre che un piacere, anche uno stile di vita. Il C è volutamente minimale e tale scelta la si riscontra anche nell'atto pratico della programmazione. Per scrivere del buon codice in C non c'è necessariamente bisogno di IDE sofisticati o altri fronzoli grafici, basta un semplice editor di testo (anche VI è perfetto!), il caro buon vecchio amico terminale ed ovviamente un compilatore.



Anche se si pensa che tanti programmi aperti siano scomodi per scrivere del codice, ci si renderà subito conto che invece questi strumenti, leggeri, sono molto pratici e svolgono perfettamente il loro dovere a differenza di un unico programma che cerca di emularli tutti.

Scegliete per bene i vostri strumenti e studiante i manuali di tali programmi per imparare ad usarli perfettamente e poter scrivere così il vostro codice senza alcun problema.

Serve davvero imparare il C?

Molti programmatori autodidatti tendono a descrivere il C come un linguaggio ormai obsoleto e sostengono che impararlo da zero è inutile e porta via semplicemente del tempo prezioso.

Non c'è nulla di più sbagliato! Il C, proprio per la potenza dei suoi strumenti e la pulizia dei suoi costrutti, è un'ottima base dalla quale partire anche per iniziare a comprendere alcuni argomenti che sono importanti indipendentemente dal linguaggio che si usa. Bisogna tener presente, in più, che molti linguaggi definiti “più moderni” sono ottenuti o vengono creati partendo proprio dal C. Il tanto blasonato C++ o anche il Java, altri non sono che discendenti del linguaggio di cui ci stiamo occupando.

Si inizia

Dopo tanti preamboli è giunto finalmente il momento di gustare un po' di codice che ci servirà per fare amicizia con il C. Come ogni guida che si rispetti anche noi inizieremo dal più classico degli “Hello World”; il nostro intento sarà proprio quello di stampare la celeberrima frase all'interno del nostro terminale.

Apriamo il nostro editor e copiamo all'interno il seguente codice:

```
#include <stdio.h>

int main(void)
{
    printf("Hello World!\n");
    return 0;
}
```

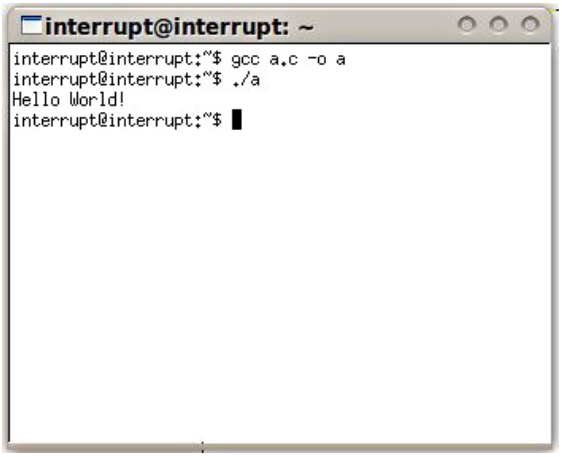
Salviamo il file con il nome `esempio.c` nella nostra Home. Rechiamoci nel terminale e compiliamo il programma digitando il comando

`gcc esempio.c -o esempio`

seguito dal tasto invio. Se il terminale non restituisce errori possiamo lanciare il nostro programma con il comando

`./esempio`

seguito da "invio". Il terminale dovrebbe presentarsi come in figura:



Analizziamo il codice:

`# include <stdio.h>`

Quando compiliamo un codice in C, esso, esegue un passaggio preliminare, ovvero viene sottoposto al precompilatore. Questo strumento si occupa di svolgere delle particolari azioni utili prima che il codice venga analizzato dal compilatore. Tutto ciò che concerne il precompilatore, viene scritto sottoforma di normale codice con la sostanziale differenza, però, che ogni rigo è preceduto dal simbolo #.

Una delle funzioni fondamentali del precompilatore è quella di includere all'interno del nostro codice le funzioni necessarie affinché esso possa funzionare. Il C infatti gode di un core davvero leggero e si basa sulla propria modularità per includere il necessario nel momento in cui se ne sente il bisogno. Dovendo richiamare la funzione `printf` diciamo al precompilatore di includere (istruzione `include`) la libreria standard `stdio.h` poiché è lì che si trova la nostra funzione. Studieremo le librerie più avanti in maniera più approfondita.

`int main()`

Ogni programma scritto in C ha bisogno di una particolare funzione che viene vista come punto di partenza dal quale il software, una volta compilato, inizia a girare. Questa funzione deve essere sempre presente e deve chiamarsi obbligatoriamente `main`. Un listato privo della funzione main non viene compilato e restituisce errore all'atto della compilazione. Come possiamo notare la parola `main` è preceduta da `int` ed è seguita dalle due parentesi tonde. Il termine `int` è un indicatore di tipo e ci indica che ogni qualvolta usciamo dalla funzione `main` per un qualsiasi motivo (errore immissione dati da parte dell'utente, fine del programma, aborto del programma, etc...) essa ci dovrà restituire necessariamente un valore intero, sia negativo che positivo. Per convenzione tale valore è indicato con 0 se il programma termina in maniera naturale e valori diversi

Per tutta la durata del corso, questa guida sarà scritta testando i codici di esempio sul seguente sistema:

Sistema: GNU/Linux Debian Squeeze
Editor: Gedit
Compilatore: Gcc
Terminale: Xterm

da 0 se si presentano altri errori.

Le parentesi, invece, servono a ricordare che main è una funzione e il loro scopo verrà precisato nelle future lezioni della guida.

`printf("Hello World!\n");`

La funzione `printf` è una delle funzioni basilari del linguaggio C. Essa ci permette di formattare particolarmente del testo e di stamparlo sull'output standard del sistema. Il suo uso è semplice ma non esclude funzionalità avanzate. In questo caso specifico abbiamo detto al linguaggio di stampare la stringa `"Hello World"` passandola come parametro alla funzione. Si noti il termine di *escape* `\n` che indica di andare a capo subito dopo la parola "World". Avremo modo di approfondire sia l'uso di printf che dei termini speciali di *escape*.

`Return 0;`

Come detto in precedenza, la funzione main deve restituire un intero quando si ha la necessità di uscire. Nel nostro caso stiamo dicendo, con il nostro codice, che abbiamo finito di fare ciò che serviva all'interno del main e possiamo uscire senza problemi. Si noti che il valore di ritorno è stato scelto come 0 per quanto detto prima. Tuttavia il suo valore è arbitrario purchè conforme al tipo dichiarato prima della funzione main.

`{}`

Le parentesi graffe svolgono nel C una funzione importante. Esse servono per delimitare graficamente (e non solo) l'inizio e la fine delle funzioni o dei blocchi logici. Omettere le parentesi provoca errore nella compilazione. Bisogna ricordarsi, pertanto, di controllare il codice in modo che ci siano tante parentesi graffe aperte quante ce ne sono chiuse.

Si noti infine che al termine delle istruzioni viene posto un punto e virgola. Esso è fondamentale e non deve essere mai dimenticato poiché indica al sistema che l'istruzione (o il blocco di istruzioni) termina in quel punto.

Ovviamente il lettore finale potrà benissimo usare comodamente il proprio sistema o un IDE se ritiene che esso possa facilitare l'apprendimento. Si tenga presente però che eventuali errori potrebbero essere dovuti al software di cui si dispone e non alla bontà del codice che si scrive.

Francesco "Ceskho OpenCode" Apruzzese

Qualora si incontrino problemi durante la lettura delle guide o se si hanno dubbi riguardanti l'argomento trattato è possibile inviare un'email a `ilrestodelpinguino@gambas-it.org`
Le risposte alle domande ritenute più interessanti verranno date nel numero successivo nella rubrica della posta in maniera tale da rendere partecipe chiunque segue il corso.

E' buona norma numerare gli errori in maniera crescente così come essi si presentano mentre si scrive il codice e creare una documentazione riguardo gli stessi corredata dal valore intero dell'errore e probabili cause.

Su che cosa e come realizzare un programma

Il computer come intelligente strumento di sussidio alle attività umane

Abbiamo visto come la nostra vita quotidiana è costellata da azioni e mezzi che si manifestano attraverso l'emissione di immagini e documenti che sono il risultato finale di un lavoro elettronico svolto da programmi diversi, ma specifici nel loro campo di esecuzione. Ma il complesso degli strumenti tecnologicamente avanzati di cui disponiamo oggi possiamo immaginarlo come un "dimensiotecno" adulto che è nato con le prime macchine tabulatrici nel lontanissimo 1889 quando l'ing. *H. Hollerith* inventò la prima di esse, a schede perforate, prototipo dei sistemi meccanografici.

Hollerith concepì un ciclo elaborativo composto da tre fasi: i dati rilevati venivano registrati su schede perforate per mezzo di una *macchina perforatrice*; le schede venivano ordinate e raggruppate secondo criteri prestabiliti (per esempio: stato di residenza, età e sesso della persona censita) da una seconda macchina, la *selezionatrice*; infine, interveniva la *tabulatrice*, la macchina cuore del sistema creato da Hollerith la quale effettuava l'elaborazione delle informazioni contenute nelle schede perforate. Essa consisteva nel sommare i dati delle varie categorie in appositi contatori, fornendo alla fine i totali ottenuti. Le prime schede perforate erano di cartoncino rettangolare della dimensione approssimativa di 215x90 mm, con fori rotondi, praticamente della stessa dimensione delle banconote statunitensi di 1 dollaro di allora. Più avanti nel tempo la loro dimensione fu ridotta a 187,325x82,55 mm, mentre i fori divennero di forma rettangolare. Cambiarono, nel tempo, pure le quantità di informazioni registrate, attraverso la perforazione. Il tipo scheda più evoluto conteneva la rappresentazione di 10 righe ed 80 colonne.

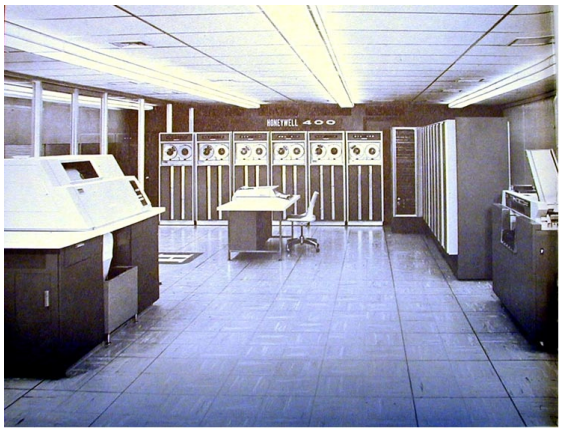
Le colonne riportavano tutte lo stesso numero, diverso per ogni riga. I numeri erano scritti in ordine di riga dall'alto verso il basso ed in successione numerica semplice; iniziavano con lo 0 e finivano col 9. Ogni numero della scheda rappresentava una "posizione". La novità concettuale era costituita dalla "presenza" o "non presenza" della perforazione in varie posizioni della scheda. La "non presenza" di foro era rappresentata in scrittura dal numero "0", mentre la sua "presenza" era rappresentata dal numero "1", secondo il sistema di numerazione binaria. Un'altra caratteristica della scheda era il taglio in diagonale dell'angolo superiore sinistro; esso rendeva obbligata la posizione della scheda da inserire nella selezionatrice, in modo da non provocare errori di lettura delle perforazioni in essa presenti. La combinazione delle perforazioni presenti in ciascuna colonna veniva tradotta automaticamente dalla macchina in un carattere alfanumerico, tramite una tabella di corrispondenza ad un codice adottato dallo stesso Hollertith, per questo, chiamato "*codice hollerith*". La selezionatrice conteneva dispositivi elottromeccanici, in grado di leggere le schede perforate e di passare poi i dati letti alla tabulatrice che era in grado, a sua volta, di svolgere operazioni aritmetiche semplici, attraverso i contatori di cui era dotata, mentre per l'esecuzione di moltiplicazioni o divisioni doveva essere collegata con un calcolatore.

Ma perché nacque la tabulatrice, col suo corredo di macchine ausiliatrici? Come la stragrande maggioranza delle invenzioni, alla sua base ci fu una necessità: quella di

stendere un censimento della popolazione da parte dell'ufficio anagrafico americano, ed il suo amministratore, J. S. Billings, ebbe l'idea di paralarne con Hollerith, allo scopo di trovare una soluzione. Quest'ultimo, che già aveva avviato lo studio e l'approntamento delle prime macchine tabulatrici, completò il suo lavoro e riuscì, nel corso di quattro anni, ad ottenere numerosi brevetti su quella macchina che poi prese il nome di tabulatrice.

L'insieme delle tre macchine, la perforatrice, la selezionatrice e la tabulatrice, era in grado di potere trattare i dati contenuti nelle schede perforate, il primo strumento di input di un sistema di elaborazione; esse diedero così vita a quello che prese il nome di "**Sistema meccanografico**". Esso progredì nel tempo e la scheda perforata divenne anche un supporto di output, per essere utilizzato come input di fasi di elaborazioni successive.

Essa regnò per tanto tempo ed i Centri meccanografici la usarono ancora a lungo. Ad essa si aggiunse anche il nastro perforato. I centri meccanografici diedero vita a varie figure professionali, dagli addetti alla perforazione, ai gestori delle selezionatrici e degli schedari adibiti alla raccolta ed alla conservazione delle "Signore Schede Perforate". Detta realtà si mantenne fino alla fase di trapasso, a partire dal 1960, quando, prima lentamente ma progressivamente, e via via sempre più velocemente la stagione della meccanografia cominciò a declinare fino a passare lo scettro del trattamento dati all'elaborazione elettronica. I vecchi centri meccanografici si trasformarono in *Centri elettronici*.



Ecco ora dobbiamo guardare proprio dentro il cambiamento, quando, con l'ingresso dell'elettronica, gli stessi centri meccanografici evolverono, perchè intervennero nuove componenti; esse (oggi le chiameremmo le "new entry") furono integrate coi vecchi strumenti, vecchi nella logica di base, ma ammodernati nel funzionamento. Si aggiunsero unità di lettura e scrittura di supporti magnetici. Ma prima ancora di dette periferiche, cominciarono ad affiorare nuove componenti, in grado di memorizzare dati e di gestirli attraverso codici elettronici. Nacquero quindi nuove figure professionali, anche se ancora la combattiva scheda perforata riuscì a svolgere il suo lavoro di principale supporto per il data entry fino agli anni '70 ed in qualche caso, anche agli inizi degli anni '80. I primi calcolatori complessi, connessi ai sistemi meccanografici, comparvero in grosse società, come Banche ed Assicurazioni, ed enti, come Università, tra il 1945 e il 1955. Erano veri e propri elaboratori elettronici ed erano costituiti per lo più da valvole termoioniche. Occupavano intere stanze ed erano lentissimi e costosissimi. Erano inaffidabili, in quanto le valvole che li componevano si rompevano spesso. La capacità di memorizzare codici e dati diede vita al "Programma". Esso veniva scritto in codice binario e trasferito, attraverso le perforatrici su schede perforate che a loro volta venivano date in pasto ai lettori di schede perforate, successori delle selezionatrici di Hollerith, per essere finalmente inseriti nella memoria dell'elaboratore, figlio della tabulatrice di Hollerith. Da quel momento l'elaboratore era pronto per ricevere i dati da trattare. Il "programma" era stato scritto da una nuova figura professionale: "il Programmatore". Ecco, finalmente abbiamo davanti le due figure principali dell'intelligenza artificiale; il programmatore con la sua intelligenza umana ed il programma, frutto dello studio del programmatore, applicato al trattamento dei dati, tramite l'elaboratore dotato della memoria elettronica dove veniva caricato il programma. La memoria dell'elaboratore, dopo avervi caricato il codice binario del programma scritto dal programmatore, costituiva e costituisce l'intelligenza artificiale. *(fine 2ª parte)*

Pietro Antonino "Picavbg" Catania

MyPaint

Risveglia l'artista che c'è in te

MyPaint è un semplice ma versatile programma di grafica focalizzato sul disegno piuttosto che sull'interfaccia.

Quanti di voi si sono mai chiesti "ma con Linux posso fare veramente di tutto"? Beh, la risposta non è difficile. Linux ha una vastità di programmi che spaziano dall'attività ludica alla gestione degli uffici, fino ai programmi per l'apprendimento e chi più ne ha più ne metta. L'unico limite: credo che non basti un semplice personal computer per contenere tutti i programmi che fornisce il nostro sistema operativo libero.

In questo numero vi voglio proporre un programma che libererà l'artista che c'è in voi e con semplici e pochi passi avrete nella vostra distribuzione preferita una tela che non avrà nulla da invidiare ad una tela artistica reale. Il programma in questione si chiama **MyPaint** ed è arrivato alla versione 0.8.2. Scarichiamo il codice sorgente e compiliamolo seguendo il *readme* nella cartella del programma.

Dipendenze richieste: pygtk, python, swig, gtk, numpy, pycairo(>=1.4), protobuf

Per gli utenti Debian possiamo utilizzare il seguente comando da terminale:

```
apt-get install g++ python-dev libglib2.0-dev python-numpy swig scons gettext python-protobuf protobuf-compiler
```

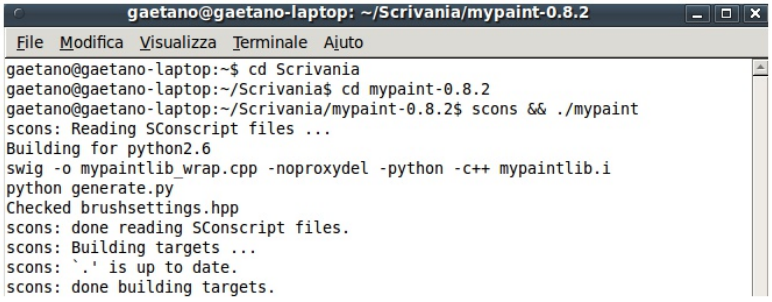
Dopo aver installato le dipendenze possiamo utilizzare il seguente script per far partire il programma:

```
scons && ./mypaint
```

In alternativa (per Ubuntu 9.10) potrete andare su questo sito e scaricare il deb già compilato e pronto:

`http://www.getdeb.net/software/MyPaint`

Il programma è molto *user friendly* e all'avvio si presenta con un foglio bianco (tela) in cui figurano i menù: File, Modifica, Visualizza, Pennello, Colore, Livelli, Aiuto.

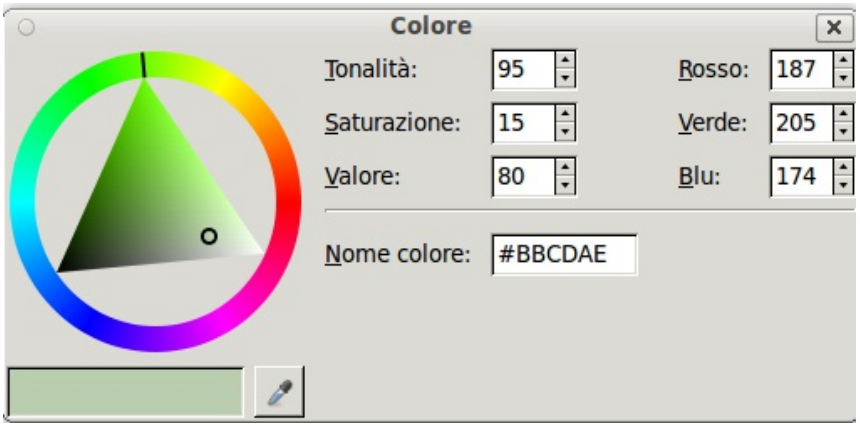


Vediamo di cosa abbiamo bisogno per iniziare a disegnare. Per comodità vi suggerisco di aprire immediatamente i pennelli di cui abbiamo bisogno cliccando su "Pennello": ci si presenterà una lista dei pennelli disponibili.

Abbiamo veramente una grande scelta che è possibile utilizzare. Tantissimi inoltre sono i diversi metodi di pittura che è possibile riprodurre spaziando dai colori ad olio, agli acrilici, ai semplici marker o ai piu comuni inchiostri. Ho trovato molto utile anche le diverse *texture* presenti. Selezionando dal menu la voce "Colore", avremo la possibilità di scegliere il colore adatto al disegno che abbiamo intenzione di creare.

Una nuova finestra di prelievo colore ci aiuterà nel caso in cui volessimo copiare in maniera veloce grazie all'uso della cannuccia. È presente anche il menu "Livello" (ancora ad uno stato embrionale), fondamentale per chi ha intenzione di creare fondendo più effetti tra loro e permettendo il salvataggio in JPEG di tutti i *layer* che pian piano si creano.

Possiamo, cliccando il tasto F11, passare alla visualizzazione a pieno schermo che trasformerà veramente il nostro PC in una tela pronta all'uso.



Con questa ultima versione è stata implementata anche la possibilità di tracciare linee dritte tramite il tasto Shift e lo zoom, che nelle vecchie versioni aveva alcuni problemi di ritardo, adesso è stato velocizzato non presentando piu particolari anomalie.

Tutte le funzionalità del programma vengono apprezzate maggiormente se si ha a

disposizione una tavoletta grafica in quanto il programma è indicato per quelle tavolette grafiche sensibili alla pressione che vi permetteranno veramente di apprezzare il programma con una semplicità stupefacente.

Potrete cambiare la dinamica del pennello nonché creare il vostro set personalizzato. Tuttavia, e ve lo consiglio vivamente, anche installandolo senza avere a disposizione una tavoletta sono sicuro che vi divertirete usando questo programma riscoprendo la vostra vena artistica.

I risultati saranno davvero strabilianti come potete vedere da questo link:

`http://wiki.mypaint.info/Galleries`

O dagli screenshot del sito ufficiale:

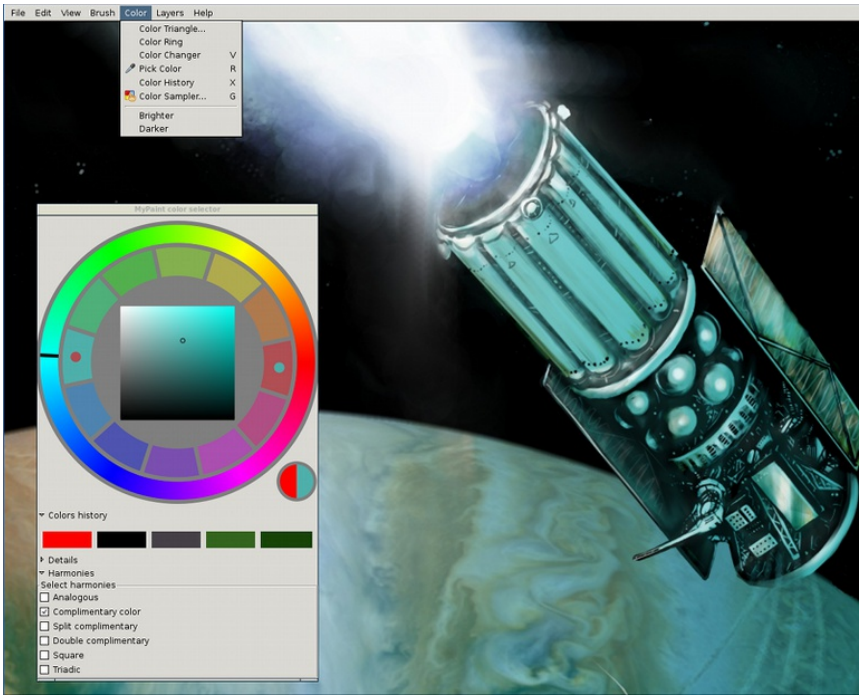
`http://mypaint.intilinux.com/?page_id=9`

Ho deciso di proporre questo programma perché è un'alternativa semplice ed immediata ai piu conosciuti Gnu Paint (ambiente Gnome) o KolourPaint (KDE) ma soprattutto perché è veramente bello da usare per esprimere al meglio la nostra voglia di creare solleticando la nostra fantasia.

Consiglio: alcuni utenti hanno trovato difficoltà a trovare il programma una volta installato. Il trucco sta nell'utilizzare lo script **`scons && ./mypaint`** a fine compilazione del programma. Alla fine il programma verrà salvato nella cartella e quando si vorrà fare partire basterà dare il comando **./mypaint** entrando nella directory tramite terminale oppure in alternativa creando un lanciatore.

Se volete avere il programma a portata di menù entrate con i permessi di root e date il seguente comando **`scons prefix=/usr/local install`**, che vi installerà l'icona nel menù del vostro sistema operativo.

Gaetano "tangoku" Lo Nigro

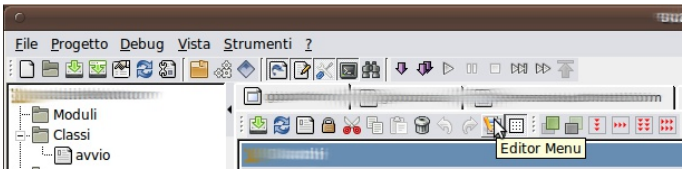


I menu con Gambas

Aggiungere i menu programma alle vostre applicazioni

Avete presente quel menu che si trova nella parte superiore di ogni programma? Bene, oggi volevo spiegare come farlo con **Gambas**.

Andate nel form dove volete aggiungere il menu e cliccate nell'icona che ha come simbolo una matita e una specie di foglio o, in alternativa, potete premere i tasti di scelta rapida CTRL+E oppure fare click con il tasto destro sul form e selezionare "*Editor Menu*".

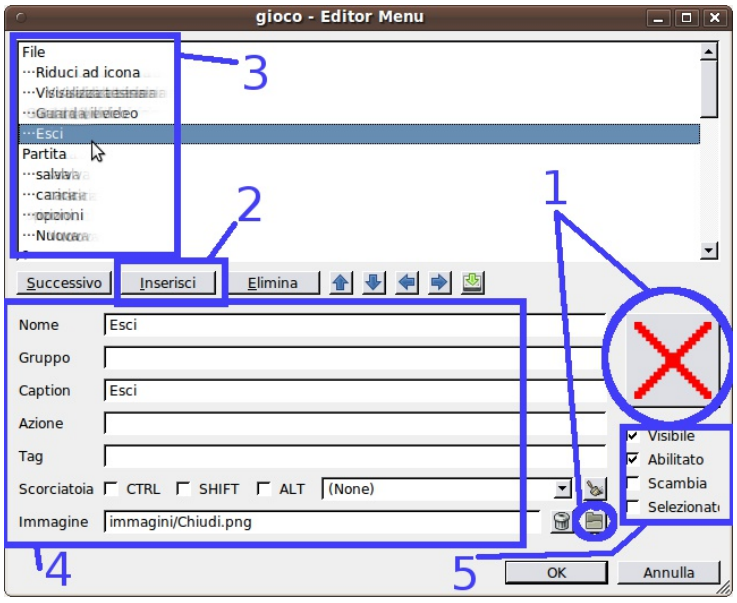


Si aprirà una finestra che vi permetterà di creare o editare il vostro menu. Cominciamo subito a inserire la prima voce. Cliccate, quindi, su "Inserisci" (2). Nella parte alta apparirà un testo, denominato "Menu1" (3), mentre in basso ci sarà l'editor per modificare il menu (4). In "Nome" potete scegliere il nome da attibuire al menu, che non sarà quello visualizzato ma bensì l'alias attribuito. Per cambiare il nome del menu visualizzato andate a modificare il campo "Caption". Nel campo "Scorciatoia" possiamo andare a modificare i tasti che, premuti da tastiera, apriranno quel menu.

A destra potete vedere una specie di pulsante vuoto: cliccatelo. Si aprirà una semplice interfaccia che vi permetterà di inserire un'immagine da associare a quel menu. Così, oltre al nome, il vostro menù avrà in parte una bella immagine. Ricordatevi di mettere lo sfondo trasparente se non volete brutte sorprese (1).

Le *checkbox* sulla destra permettono di scegliere se abilitare il menu oppure disabilitarlo, cioè renderlo non cliccabile, così il testo apparirà di un grigio chiaro. Potete anche scegliere di renderlo visibile o meno. Se non lo spuntate quella voce non si vedrà (5). Tutti gli altri campi per ora sono superflui. Provate a cliccare su "Elimina": così facendo la voce del menu selezionata verrà eliminata.

Ora seguendo quello che ho scritto sopra create 4-5 menu: si disporranno tutti in linea sul vostro form. Ma se volessimo creare dei sottomenu? Basterà selezionare la voce da mettere come sottomenu di un'altra, spostarla con le frecce che si trovano al centro sotto la voce del menu dove vogliamo che appaia e infine premere la freccia destra, che si trova sempre al centro. Infatti con le frecce su e giù riordiniamo le voci del menu, mentre con le frecce destra e sinistra aumentiamo o scaliamo di livello, quindi mettiamo quella voce come sottomenu o meno. Sarà possibile creare sottomenu di sottomenu, molto semplicemente facendo le stesse operazioni dette prima. Ora provate ad avviare il programma e a vedere il risultato.



Per assegnare una procedura (*subroutine*) o una funzione (*function*) ad una voce del menu basterà andarci sopra quando ci troviamo nel form e cliccarci. Oppure scrivere `PUBLIC SUB nome_click()` dove "nome" sta per il nome che avete dato al menu (non l'attributo *caption*, ma proprio il nome). Alcuni attributi molto importanti e utilizzati del menu sono: *hide*, *show*, *enabled*.

Basterà scrivere `nome.hide` per nascondere il menu e `nome.show` per mostrare il menu. Molto utile nel caso volessimo mostrare un menu solo a certe condizioni. Nel caso non volessimo nasconderlo ma solo mostrarlo disabilitato basterà andare a modificare l'attributo *enabled*: `nome.enabled = FALSE`. Così facendo non sarà possibile cliccare nel nostro menu, seppure si possa vedere. Invece scrivendo `nome.enabled = TRUE` il nostro menu sarà di nuovo attivato, quindi sarà possibile cliccarlo. Ormai conoscete le caratteristiche principali del menu, ora tocca a voi scoprire qualcosa in più.

Alessio "Ealmuno" Moretto

Solo le voci del menu possono essere associate ad un'azione, non il menu principale e neanche i sottomenu che contengono altre voci

La funzione Replace()

Come sostituire parti di testo all'interno delle stringhe

Replace() è una funzione nativa di Gambas che permette di sostituire "al volo" sezioni della stringa passata come argomento con il contenuto di un'altra stringa.

Se vi ricordate, nello scorso numero della rivista abbiamo parlato dell'uso della funzione *Split()*: l'articolo finiva facendo riferimento ad un'altra funzione che avrebbe preso in carico l'output finale della funzione per filtrarlo definitivamente.

Alla fine dell'articolo con la funzione Split avevamo ottenuto, infatti, un *array* che, al suo interno, presentava stringhe di vario contenuto e lunghezza diversa, nelle quali però erano rimasti ancora degli apostrofi (carattere '), estranei alle stringhe effettive necessarie al programma. Occorre pertanto svolgere un ulteriore passo filtrando ciascuna di esse ed eliminando tutti i caratteri non significativi.

La funzione in grado di svolgere tale compito in maniera semplice e snella è la funzione **Replace()**:

```
Risultato = Replace (Stringa , Motivo , StringaSostitutiva [, Comparazione ])
```

dove `Risultato` contiene l'esito relativo all'esecuzione della funzione:

0 qualora il valore contenuto in `Motivo` sia presente in `Stringa`;

-1 qualora il valore contenuto in `Motivo` non sia presente in `Stringa`;

`Stringa` è la stringa dove la funzione svolge la ricerca del/dei carattere/i da sostituire; `Motivo` è la sottostringa di `Stringa` il cui valore, se trovato, va sostituito col valore contenuto in `StringaSostitutiva`, che è la stringa contenente il valore che va a rimpiazzare in `Stringa` ciascuna ricorrenza equivalente al contenuto di `Motivo`; `Comparazione` rappresenta il tipo di confronto in forma di campo numerico *integer*, e può contenere:

"0" (*gb.Binary*): di tipo binario (è quello usato di *default*);

"1" (*gb.Text* o *gb.Case*): di tipo testo, non sensibile al maiuscolo/minuscolo.

La descrizione indicata sopra corrisponde con quanto riportato nella documentazione di Gambas (`http://gambasdoc.org/help/lang/replace`)

Nell'uso pratico della funzione essa si usa nel seguente formato:

```
Stringa=Replace (stringa-di-partenza, stringa-da-trovare, stringa-di-sostituzione)
```

Per fare un esempio, supponiamo di dovere sostituire nella stringa, primo termine di confronto, "$Contenitore", la sottostringa "$Cercato", con la costante "$Nuovo"

```
Dim $Contenitore as String = "Istruzioni per l'uso"

Dim $Cercato as String = "uso"

Dim $Nuovo as String = "utente"

  $Contenitore = Replace ($Contenitore, $Cercato, $Nuovo)
```

Dopo l'esecuzione della funzione il campo $Contenitore conterrà il seguente valore: "Istruzioni per l'utente"

Supponiamo ora che la costante $Nuovo contenga la stringa "pronte"; allora il codice

```
  $Contenitore = Replace ($Contenitore, $Cercato, $Nuovo)
```

non produrrà alcun effetto, perché la parola "pronte" non è presente nella stringa $Contenitore; allora, dopo l'esecuzione della funzione, il campo $Contenitore conterrà sempre il valore iniziale: "Istruzioni per l'uso".

Se però volessimo essere sicuri di effettuare la sostituzione in modo da prevenire condizioni che possano produrre un comportamento non desiderato nella sequenza delle istruzioni del programma, potremmo pensare al codice seguente:

```
Esito = ($Contenitore = Replace ($Contenitore, $Cercato, $Nuovo))

IF Esito = 0 THEN

   $Contenitore = Replace ($Contenitore, $Cercato, $Nuovo)

ELSE

   PRINT "contenuto di $Nuovo ERRATO:" & $Nuovo

ENDIF
```

Tornando all'*array* che era stato formato nella parte conclusiva dell'argomento sulla funzione Split(), vediamo come applicare la funzione Replace() per pulire le stringhe di ciascun elemento dell'*array* dalla presenza

dei caratteri non pertinenti alle stringhe costituenti gli elementi dell'*array*. L'*array* ottenuto era:

```
DIM $RgRecDB AS NEW String[]
```

col seguente contenuto:

```
----------------------1° elemento
(10100000
'CASA'
'N'
nullo)
----------------------2° elemento
(10100100
'Emollienti'
'N'
nullo)
----------------------3° elemento
(10100102
'Pensione gallo'
'N'
nullo)
----------------------4° elemento
(10100600
'Riscossa avantitutta'
'S'
nullo)
----------------------5° elemento
(10100700
'Vittoria'
'N'
nullo)
```

Le nuove istruzioni da svolgere sono:

```
FOR iInd = 0 TO iTotElRg
      $RgRecDB[iInd] =
Replace($RgRecDB[iInd], "(", "")
'Elimina dalla stringa tutti i crt "("
      $RgRecDB[iInd] =
Replace($RgRecDB[iInd], "'", "")
'Elimina dalla stringa tutti i crt
“apice”
      $RgRecDB[iInd] =
Replace($RgRecDB[iInd], ")", "")
'Elimina dalla stringa tutti i crt ")"
NEXT
```

Tutte le istruzioni Replace agiscono soltanto sugli elementi contenenti la stringa cercata e, una volta trovata, sostituiscono con il carattere vuoto la sottostringa cercata, vale a dire: "**(**" nella prima Replace, "'" nella seconda "Replace", "**)**" nella terza Replace.

In pratica eliminano il carattere concettualmente estraneo alla stringa. Il metodo permette così di ottenere una sequenza di elementi nettati da qualsiasi carattere estraneo all'effetivo loro contenuto. Pertanto, l'*array* $RgRecDB contiene:

```
----------------------1° elemento
10100000
CASA
N
nullo
----------------------2° elemento
10100100
Emollienti
N
nullo
----------------------3° elemento
10100102
Pensione gallo
N
nullo
----------------------4° elemento
10100600
Riscossa avantitutta
S
nullo
----------------------5° elemento
10100700
Vittoria
N
nullo
```

pronti per essere elaborati più avanti.

Pietro Antonino "picavbg" Catania

Recensioni

C - Didattica e programmazione

Un libro di facile lettura per imparare i rudimenti del C

Quando si inizia a studiare un argomento nuovo, al di là del suo campo di appartenenza, il primo interrogativo che ci si pone è quali siano i migliori strumenti per cominciare a studiare. L'informatica, ovviamente, non fa eccezione ed in particolare la programmazione prevede che l'utente abbia per le mani una buona guida che lo conduca passo passo verso la conoscenza più o meno sommaria degli strumenti con i quali poi si troverà a lavorare. La scelta della guida a cui affidarsi è importante e la struttura e il contenuto della stessa saranno il collante che terrà l'apprendista programmatore attaccato o meno alla serie per la smania di sviluppare codice su codice.

La recensione di oggi pone in bacheca un libro davvero molto ben fatto e dalla struttura semplice e perfettamente adattata alla didattica per chi vuole intraprendere l'avventura della programmazione in linguaggio C.

"*C - Didattica e programmazione*", scritto da Al Kelley e Ira Pohl e pubblicato nel 2004, è un libro al limite tra la guida per chi sa già programmare e il testo didattico per chi vuole imparare. La struttura degli argomenti, infatti, prevede la conoscenza del linguaggio dalle basi più semplici alla gestione dei dati più avanzata. Si inizia con il semplice "Hello World", per comprendere uso di librerie e funzioni, sino a giungere all'interazione tra codice C e sistema operativo, per modificare i permessi ai file o per sfruttare la riga di comando.

La sua composizione si sviluppa su 652 pagine che seguono il seguente schema:

CAP 0: Partire da zero
CAP 1: Panoramica sul C
CAP 2: Elementi lessicali, operatori e sistema C
CAP 3: Tipi di dati fondamentali
CAP 4: Flusso del controllo
CAP 5: Funzioni
CAP 6: Array, puntatori e stringhe
CAP 7: Operatori orientati ai bit e tipi enumerativi
CAP 8: Il preprocessore
CAP 9: Strutture e unioni
CAP 10: Strutture e trattamento liste
CAP 11: Input/output e sistema operativo
CAP 12: Applicazioni avanzate
CAP 13: Dal C al C++
CAP 14: Dal C al Java

APP A: La libreria standard
APP B: Sintassi del linguaggio C
APP C: Confronto tra linguaggio ANSI C e C tradizionale
APP D: Codifiche ASCII
APP E: Priorità e associatività degli operatori

È importante notare come gli argomenti del libro non siano strutturati in modo che essa sia una guida ma piuttosto si fa sì che essa accompagni i neofiti attraverso il difficile mondo del C. Le appendici finali sono importantissime una volta che si è terminato il percorso didattico dell'opera in modo da trovare le informazioni utili in maniera veloce senza ripercorrere nuovamente tutte le pagine interne.

Gli argomenti sono trattati mediante la lettura di esempi pratici che inseriscono la parte teorica in contesti pratici che evidenziano gli usi che i diversi elementi possono avere nei casi reali in cui ci si può imbattere. Ogni riga di codice è ben commentata sia dai classici commenti interni che dalle trattazioni teoriche. Notevole è, infine, la volonta degli autori di confrontare il C con due altri linguaggi derivati, quali il C++ e il Java, in modo che il lettore si renda conto delle differenti strutture di programmazione che si possono incontrare e delle sostanziali differenze tra lingauggio e linguaggio.

Il libro è reperibile nei maggiori rivenditori on-line di prodotti editoriali ed è sicuramente un'acquisto consigliato per chi si vuole cimentare avendo come supporto un ottimo prodotto. È possibile visionare un'anteprima del contentuo del libro mediante l'apposita pagina di Google Libri.

Francesco "Ceskho Open Code" Apruzzese

BOX VOTI (su 5)	
Contenuto	****
Struttura	*****
Esempi	*****
Facilità	****
Chiarezza	****

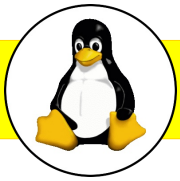
Titolo:
C - Didattica e programmazione

Autori:
Al Kelley, Ira Pohl

Editore:
Pearson Education Italia - 2004

Pagine: 652

Prezzo: € 39,00



La cattedrale ed il bazar

Due stili di sviluppo del software a confronto: il modello "cattedrale" e quello "bazar"

2. La posta deve passare.

Dal 1993 mi occupo del lato tecnico di un piccolo provider Internet gratuito chiamato Chester County InterLink (CCIL) in West Chester, Pennsylvania (sono tra i fondatori di CCIL e autore del software specifico per il nostro bulletin-board multiutenti – si può dare un'occhiata facendo telnet su locke.ccil.org: `telnet://locke.ccil.org`. Ora dà accesso a quasi tremila utenti su trenta linee). Grazie a questo lavoro posso collegarmi a Internet per 24 ore al giorno con una linea a 56K di CCIL – in realtà, è proprio quel che mi viene richiesto!

Di conseguenza sono ormai abituato alle email istantanee. Per vari motivi, era difficile far funzionare la connessione SLIP tra la mia macchina a casa (snark.thyrsus.com) e CCIL. Quando finalmente ci sono riuscito, mi dava fastidio dover fare ogni tanto telnet su locke per controllare la posta. Volevo fare in modo che i messaggi arrivassero direttamente su snark così da esserne tempestivamente avvisato e poterli gestire a livello locale.

Il semplice “sendmail forwarding” non avrebbe funzionato, perché la mia macchina personale non è sempre online e non ha un indirizzo IP statico. Mi serviva un programma in grado di raggiungere la connessione SLIP e tirar via la posta per farla arrivare localmente. Sapevo dell'esistenza di simili cose, e del fatto che in genere facevano uso di un semplice protocollo noto come POP (Post Office Protocol). E sicuramente doveva già esserci un server POP3 incluso nel sistema operativo BSD/OS di locke.

Mi serviva un client POP3. Ne ho localizzato subito uno online. Anzi, ne ho trovati tre o quattro. Per un po' ho usato un pop-perl, ma era privo di quella che pareva una funzione ovvia, la capacità di effettuare un “hacking degli indirizzi della posta prelevata in modo che il reply funzionasse correttamente.

Questo il problema: supponiamo di ricevere un messaggio da qualcuno di nome 'joe' su locke. Se lo inoltro su snark e poi cerco di fare reply, il mio programma di posta proverebbe simpaticamente a inviarlo a un inesistente 'joe' su snark. Modificare a mano ogni indirizzo per aggiungere “@ccil.org” diventerebbe in un attimo un problema serio.

Chiaramente questa era un'operazione che toccava fare al computer per conto mio. Ma nessuno dei client POP esistenti sapeva come! E questo ci porta alla prima lezione:

1. Ogni buon lavoro software inizia dalla frenesia personale di uno sviluppatore.

Forse ciò avrebbe dovuto risultare ovvio (è risaputo da tempo che “la necessità è la madre di tutte le invenzioni”), ma troppo spesso gli sviluppatori trascorrono le giornate impegnati a guadagnarsi da vivere con programmi di cui non hanno alcun bisogno e che non apprezzano. Ma non nel mondo Linux – il che spiega l'alta qualità media del software originato dalla comunità Linux.

Mi sono forse lanciato in un'attività frenetica per scrivere il codice di un client POP3 nuovo di zecca in grado di competere con quelli esistenti? Nemmeno per sogno! Ho esaminato attentamente le utility POP che avevo in mano, chiedendomi: “qual'è la più vicina a quel che sto cercando?” Perché:

2. I bravi programmatori sanno cosa scrivere. I migliori sanno cosa riscrivere (e riusare).

Pur non ritenendomi un programmatore tra i più bravi, cerco di imitarli. Importante caratteristica di costoro è una sorta di ozio costruttivo. Sanno che si ottiene il meglio non per le energie impiegate ma per il risultato raggiunto, e che quasi sempre è più facile iniziare da una buona soluzione parziale piuttosto che dal nulla assoluto.

Linus Torvalds, per esempio, non ha mai cercato di riscrivere Linux da zero. È invece partito riutilizzando codici e idee riprese da Minix, piccolo sistema operativo per macchine 386 assai simile a Unix. Alla fine il codice Minix è scomparso oppure è stato completamente riscritto – ma per il tempo che è rimasto lì presente è servito come impalcatura per l'infante che sarebbe infine divenuto Linux.

Con lo stesso spirito, mi sono messo a cercare una utility POP basata su codici ragionevolmente ben fatti, da utilizzare come base di sviluppo.

La tradizione di condivisione dei codici tipica del mondo Unix ha sempre favorito il riutilizzo dei sorgenti (questo il motivo per cui il progetto GNU ha scelto come sistema operativo di base proprio Unix, nonostante alcune serie riserve sullo stesso). Il mondo Linux ha spinto questa tradizione vicina al suo al limite tecnologico; sono generalmente disponibili terabyte di codice open source.

L'autore, *Eric Steven Raymond*, è un informatico statunitense nato nel 1957. E' lo sviluppatore del software **fetchmail**, è l'attuale manutentore del **Jargon File** (meglio noto come "Il nuovo dizionario degli hacker") e l'autore del **Glider**, il simbolo identificativo degli hacker. E' una figura prominente del panorama hacker internazionale e nel 1997 ha pubblicato un'analisi su due stili differenti di sviluppo del software libero. Il saggio, intitolato "**La cattedrale e il bazaar**", è generalmente considerato il manifesto del movimento open source.

Il resto del Pinguino pubblicherà il saggio a dispense per motivi di lunghezza dello stesso.

Il saggio è liberamente consultabile su WikiSource all'indirizzo: `http://it.wikisource.org/wiki/La_cattedrale_e_il_bazaar`

È quindi probabile che, impiegando del tempo a cercare il lavoro di qualcuno quasi ben fatto, si ottengano i risultati voluti. E ciò vale assai più nel mondo Linux che altrove.

Proprio quel che è successo a me. Conteggiando i programmi trovati prima, con la seconda ricerca ottenni un totale di nove candidati – fetchpop, PopTart, get-mail, gwpop, pimp, pop-perl, popc, popmail e upop. Il primo su cui mi sono concentrato è stato 'fetchpop' di Seung-Hong Oh. Ho inserito l'opzione “header-rewrite” e ho apportato altri miglioramenti, poi accettati dall'autore nella release 1.9.

Alcune settimane dopo, però, mi sono imbattuto nel codice di 'popclient' scritto da Carl Harris, ed mi sono trovato di fronte a un problema. Pur offrendo alcune buone idee originali (come la modalità “daemon”), fetchpop poteva gestire solo POP3 e il codice rifletteva un certo approccio da dilettante (Seung-Hong era un programmatore brillante ma inesperto, ed entrambe le qualità risultavano evidenti). Il codice di Carl era migliore, alquanto professionale e solido, ma il suo programma difettava di varie opzioni presenti in fetchpop, opzioni importanti e piuttosto complesse da implementare (incluse quelle aggiunte dal sottoscritto).

Restare o cambiare? Nel secondo caso avrei buttato via il codice che avevo già scritto in cambio di una migliore base di sviluppo.

Un motivo pratico per passare all'altro programma era il supporto per protocolli multipli. POP3 è il più usato tra i server per l'ufficio postale, ma non è il solo. Fetchpop e l'altro rivale non avevano POP2, RPOP, o APOP, e io stesso stavo meditando, giusto per divertimento, l'aggiunta di IMAP (Internet Message Access Protocol, il protocollo per l'ufficio postale più recente e più potente).

Avevo però altri motivi teorici per ritenere una buona idea il fatto di cambiare, qualcosa che avevo imparato molto tempo prima di Linux.

3. “Preparati a buttarne via uno; dovrai farlo comunque.” (Fred Brooks, “The Mythical Man-Month”, Capitolo 11)

In altri termini, spesso non si riesce a comprendere davvero un problema fino alla prima volta in cui si prova a implementarne la soluzione. La seconda volta forse se ne sa abbastanza per riuscirci. Per arrivare alla soluzione, preparati a ricominciare almeno una volta.

Be', mi son detto, la mia prima volta erano state le modifiche a fetchpop. Adesso era ora di cambiare, e così feci.

Dopo aver mandato a Carl Harris il 25 Giugno 1996 una prima serie di aggiustamenti per popclient, mi resi conto che da qualche tempo egli aveva perso interesse nel programma. Il codice era un po' polveroso, con vari bug in giro. Avrei dovuto fare molte modifiche, e ci mettemmo rapidamente d'accordo sul fatto che la cosa più logica fosse che il programma passasse in mano mia.

Senza che me ne accorgessi più di tanto, il progetto era cresciuto parecchio. Non mi stavo più occupando soltanto di sistemare i piccoli difetti di un client POP già esistente. Mi ero addossato l'intera gestione di un programma, e mi venivano in mente delle idee che avrebbero probabilmente portato a modifiche radicali.

In una cultura del software che incoraggia la condivisione del codice, non si trattava altro che della naturale evoluzione di un progetto. Questi i punti-chiave:

4. Se hai l'atteggiamento giusto, saranno i problemi interessanti a trovare te.

Ma l'atteggiamento di Carl Harris risultò perfino più importante. Fu lui a comprendere che:

5. Quando hai perso interesse in un programma, l'ultimo tuo dovere è passarlo a un successore competente.

Senza neppure parlarne, io e Carl sapevamo di perseguire il comune obiettivo di voler raggiungere la soluzione migliore. L'unica questione per entrambi era stabilire se le mie fossero mani fidate. Una volta concordato su questo, egli agì con gentilezza e prontezza. Spero di comportarmi altrettanto bene quando verrà il mio turno.

(Continua...)

Gli autori

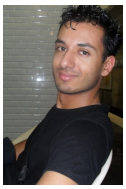
Elenco degli autori che hanno contribuito alla stesura degli articoli pubblicati su questo numero

Fabio "Pixel" Colinelli

Appassionato di informatica è uno dei moderatori del Forum di Ubuntu-it. Programmatore e fumettista a tempo perso è co-fondatore del portale italiano di Gambas.

Francesco "Ceskho OpenCode" Apruzzese

Appassionato di tutto ciò che ha un processore. Ama programmare in qualsiasi linguaggio esista. Troppo intelligente per frequentare l'università. Il suo sogno è hackerare il proprio computer senza che lui stesso se ne accorga.



Gaetano "tangoku7" Lo Nigro

Sono un segretario d'albergo con la passione per l'informatica. Lavoro all'estero come receptionist/night auditor. Mi piace scrivere e fare piccoli lavoretti grafici utilizzando solo software opensource. Mi piace stare a contatto con le persone e amo tenermi informato su tutto quello che succede nel mondo.

Pietro Antonino "picavbg" Catania

Sin da bambino ho avuto attrazione per le macchine calcolatrici e poi, più grandicello, per l'informatica. Diplomato in Ragioneria ho potuto, dopo alcuni anni, affacciarmi nel mondo del lavoro e lì, alla prima occasione, ho partecipato ad un test attitudinale di informatica che ho superato; da quel momento, ottenuta la qualifica di programmatore, ho potuto rapportarmi per quasi tutta la vita lavorativa con gli elaboratori elettronici e coi computer.

Alessio "ealmuno" Moretto

Frequento l'ITIS Informatica, amo i computer e tutto quello che c'entra con la tecnologia e il suo sviluppo, in particolare la programmazione e un po' di grafica.

Per contattare uno degli autori in merito ad un suo articolo (spiegazioni, approdonfimenti, correzioni, ecc...) potete scrivere a `ilrestodelpinguino@gambas-it.org` specificando l'articolo in questione ed il numero su cui è comparso il pezzo.

"Il resto del Pinguino" è la rivista digitale ufficiale di **Gambas-it.org**, la comunità italiana dei programmatori Gambas.
Pubblicazione aperiodica

Tutti i testi e le immagini contenuti in questa rivista sono rilasciati sotto la licenza **Creative Commons Attribuzione - Non commerciale - Condividi allo stesso modo 2.5** (vedi sotto). Ciò significa che siete liberi di adattare, copiare, distribuire ed inviare gli articoli solo alle seguenti condizioni: la paternità dell'opera deve essere attribuita in qualsiasi modo (con almeno un nome, un'e-mail o un URL) all'autore originale e al nome di questa rivista digitale ("Il resto del Pinguino") e all'URL www.gambas-it.org (ma non si possono attribuire il o gli articoli in alcun modo che lasci intendere che gli autori e la rivista abbiano esplicitamente autorizzato voi o l'uso che fate dell'opera). Se alterate, trasformate, o aggiungete informazioni all'opera, dovete distribuire il lavoro risultante con la stessa licenza o una simile o compatibile.

creative commons

Attribuzione-Non commerciale-Condividi allo stesso modo 2.5 Italia

Tu sei libero:

di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire e recitare quest'opera

di modificare quest'opera

Alle seguenti condizioni:

Attribuzione — Devi attribuire la paternità dell'opera nei modi indicati dall'autore o da chi ti ha dato l'opera in licenza e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.

Non commerciale — Non puoi usare quest'opera per fini commerciali.

Condividi allo stesso modo — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.

Prendendo atto che:

* **Rinuncia** — E' possibile rinunciare a qualunque delle condizioni sopra descritte se ottieni l'autorizzazione dal detentore dei diritti.

* **Pubblico Dominio** — Nel caso in cui l'opera o qualunque delle sue componenti siano nel pubblico dominio secondo la legge vigente, tale condizione non è in alcun modo modificata dalla licenza.

* **Altri Diritti** — La licenza non ha effetto in nessun modo sui seguenti diritti:
  - o Le eccezioni, libere utilizzazioni e le altre utilizzazioni consentite dalla legge sul diritto d'autore;
  - o I diritti morali dell'autore;
  - o Diritti che altre persone possono avere sia sull'opera stessa che su come l'opera viene utilizzata, come il diritto all'immagine o alla tutela dei dati personali.

* **Nota** — Ogni volta che usi o distribuisci quest'opera, devi farlo secondo i termini di questa licenza, che va comunicata con chiarezza.

La copia completa della licenza è reperibile a questo indirizzo:
`http://creativecommons.org/licenses/by-nc-sa/2.5/it/legalcode`