AmigaBasic: i comandi grafici

di Andrea de Prisco

Secondo appuntamento con l'AmigaBasic. Questo mese mostreremo l'uso della «complessa» grafica di Amiga attraverso i comandi disponibili nel Basic fornito con la macchina. Proprio in previsione di quest'articolo, sul numero 63 di MC (maggio) sono state già discusse tali tematiche, in quella sede senza fare specifici riferimenti a linguaggi di programmazione. In queste pagine, contrariamente, ignoreremo i particolari implementativi rimandandovi all'articolo precedente se necessario.

Grafica e testo

Iniziamo subito col dire che in Amiga non esiste il concetto di pagina testo e/o pagina grafica, ma si parla sempre e solo di schermi e finestre. Gli output grafici di testo avvengono solo ed esclusivamente nelle finestre le quali «appartengono» agli schermi. Più schermi possono essere visualizzati, sfruttando per ognuno di essi la risoluzione e i colori desiderati. Come per le finestre anche gli schermi hanno priorità e per far apparire o scomparire uno di questi si clicka sugli appositi gadget in alto a sinistra o si aggancia col mouse la drag-bar superiore per effettuare spostamenti in senso verticale.

Tutte le finestre di uno schermo, hanno la medesima risoluzione e uguale palette di colori: per visualizzare risoluzioni diverse occorre definire due schermi diversi ed aprire in ognuno una finestra coi colori desiderati.

Detto questo, una volta attivata una finestra potremo scrivere o disegnare su questa indifferentemente. Possiamo tracciare una linea col comando LINE, un cerchio col comando CIRCLE e stampare caratteri col comando PRINT. Lo schermo di default è quello del Workbench (quindi 640 × 200 in 4 colori) mentre la finestra di default è quella del Basic: un PRINT o un comando grafico dato «liscio» agirà proprio in quella.

Grafica «di base»

Prima di parlarvi più approfonditamente di schermi e finestre, iniziamo subito con i comandi grafici di base: punti, linee, rettangoli, cerchi e riempimento di aree.

Tutte le coordinate possono essere assolute (origine dei punti nell'angolo in alto a sinistra della finestra) oppure relative all'ultimo punto tracciato. Una coordinata si indica raccogliendo tra parentesi l'ascissa e l'ordinata (separate da virgola es.: (300,125)) eventualmente prefissandola con la parola chiave STEP se si tratta di coordinate relative. Il comando per pulire una finestra è CLS, e come tutti i comandi si riferisce alla finestra attiva in quel momento. Disegniamo un punto nella finestra.

```
PSET (200,160)
```

Per default, il colore del punto sarà quello dei caratteri (standard bianco). Se vogliamo disegnare un punto di un altro colore indicheremo il suo numero (posizione nella palette) di seguito alle coordinate. Ad esempio:

```
PSET (400,50),3
```

Per completezza ricordiamo che lo schermo del Workbench è a 4 colori quindi in questo caso potremo specificare un colore compreso tra 0 a 3.

Se desideriamo tracciare un punto 10 pixel più a destra del precedente useremo le coordinate relative:

```
PSET STEP (10,0)
```

Il primo valore tra parentesi è lo spostamento di ascissa, il secondo quello di ordinata: (10,0) sta proprio per «10 pixel più a destra».

Linee: discorso analogo. Indichiamo i due punti da unire con una linea. Ad esempio, per tracciare una linea tra i punti (0,0) e (200,100) scriveremo:

```
LINE (0,0)-(200,100)
```

Come per i punti possiamo adoperare le coordinate relative all'ultimo punto tracciato e/o indicare il colore. Omettendo la prima coordinata (ma non il trattino) tracceremo una linea dall'ultimo punto tracciato fino al punto indicato. Oltre a questo, con la stessa istruzione possiamo tracciare rettangoli vuoti o pieni indicando, dopo il colore, la chiave B o BF (Box o Box Fill). Disegnamo un quadrato (forma dei pixel a parte!):

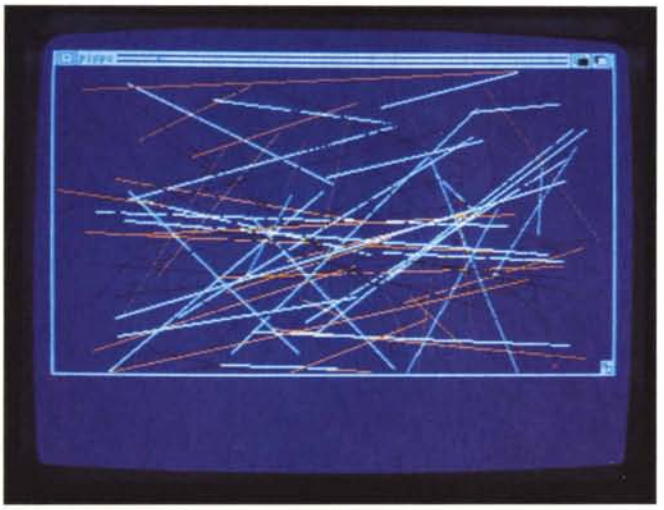
```
LINE (10,10)-STEP (100,100),3,BF
```

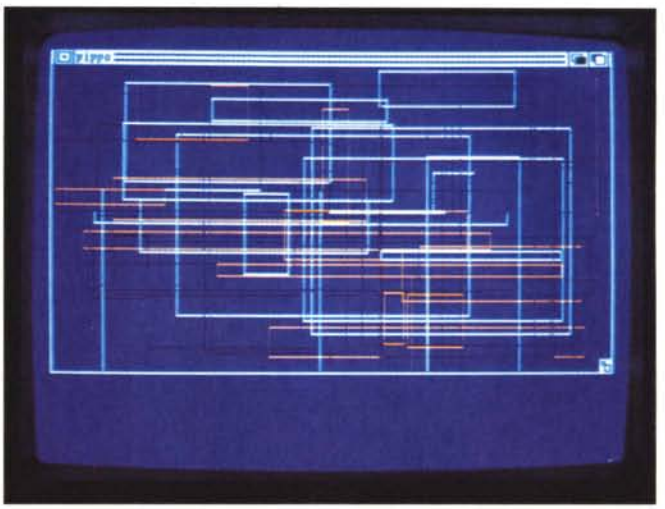
Il quadrato tracciato sarà largo 100 pixel posizionato a partire dal punto (10,10).

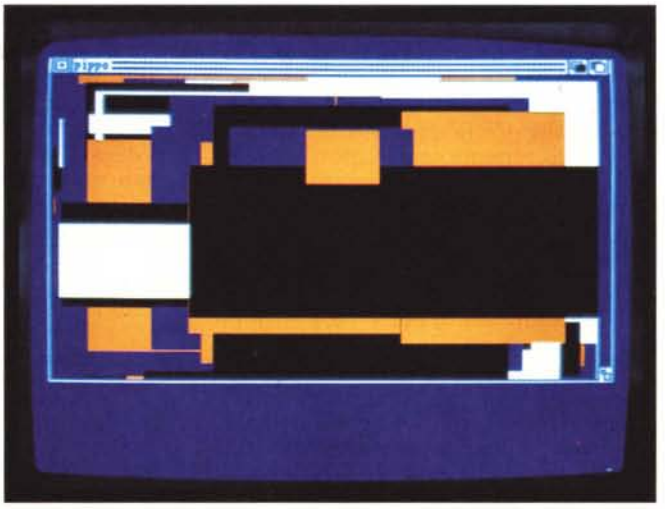
Per tracciare un cerchio, si utilizza il comando CIRCLE, che permette di disegnare anche ellissi e archi. La sintassi del comando è:

```
CIRCLE ce,ra,co,in,fi,as
```

«ce» sta per coordinate del centro, come al solito assolute o relative (prefisso STEP). Segue il raggio espresso in pixel e il colore, come prima, uno della palette, «in» e «fi» indicano inizio e fine arco, espresso in radianti, secondo il normale orientamento trigonometrico. Infine «as» indica l'aspetto, da usarsi, per gli ellissi o per i cer-







Col comando LINE è possibile tracciare linee, rettangoli vuoti o pieni.

chi se il nostro monitor non è tarato bene (sul retro del monitor sono presenti gli opportuni regolatori).

Qualche esempio:

```
CIRCLE (100,120), 40
CIRCLE (50,70),100,3
CIRCLE (80,100),60,,,,2
```

il primo traccia un cerchio di raggio 40 colore default, il secondo un cerchio raggio 100 colore 3. Il terzo un'ellisse (notare le 4 virgole di seguito che delimitano i parametri «non passati»).

Per finire questo primo gruppo, il comando PAINT permette di riempire aree delimitate dal colore specificato (ahimé). Indicheremo il punto da cui iniziare il riempimento (assoluto o relativo) il colore con cui «dipingere» e il colore del bordo dell'area da riempire. Ad esempio:

```
PAINT (100,100),2,3
```

colora di «2» l'area attorno al punto (100,100) delimitata dal contorno color «3». Semplice, no?

Finestre e schermi

Per aprire una nuova finestra nello schermo è disponibile il comando WINDOW. Lo stesso è utilizzato anche per chiudere una finestra, renderla attiva e/o prioritaria rispetto alle altre («emerge»).

Ogni finestra ha un numero identificatore, in modo da poterla riferire anche dopo la sua apertura. Potremo indicare dove far apparire la window, eventualmente mostrando un titolo, con quale dimensione e con quali gadget. La sintassi è molto semplice:

```
WINDOW id,ti,re,ty,sc
```

id, come detto sarà l'identificatore: un numero qualsiasi. Segue il titolo della finestra racchiuso tra apici, le dimensioni e posizione della finestra indicano la diagonale del rettangolo di schermo che occuperà, il tipo (ne parleremo tra poco) e a quale schermo ci riferiamo, se ne abbiamo definito altri. Ad esempio, per aprire una finestra scriveremo qualcosa come:

```
WINDOW 2, «titolo», (10,10)-(300,100)
```

Notare la sintassi della posizione, analoga a quella del tracciamento di una linea o un rettangolo.

Per quanto riguarda il parametro «ty», tipo, esso è un numero compreso tra 0 e 31 risultante dalla somma delle costanti corrispondenti ai vari gadget richiesti. La costante per la «dimensione variabile» è 1, «finestra spostabile» è 2, «priorità variabile» è 4 (notare come siano tutte potenze di 2), «finestra chiudibile» è 8, se desideriamo l'autorefresh 16. Ad esempio per aprire una finestra spostabile e chiudibile il ty sarà 10 (=2+8). Una finestra completa ha ty pari a 31 (default), una finestra senza alcun gadget ha ty pari a 0.

Per chiudere una finestra useremo il comando:

```
WINDOW CLOSE id
```

per attivarla per l'output facendola «emergere»:

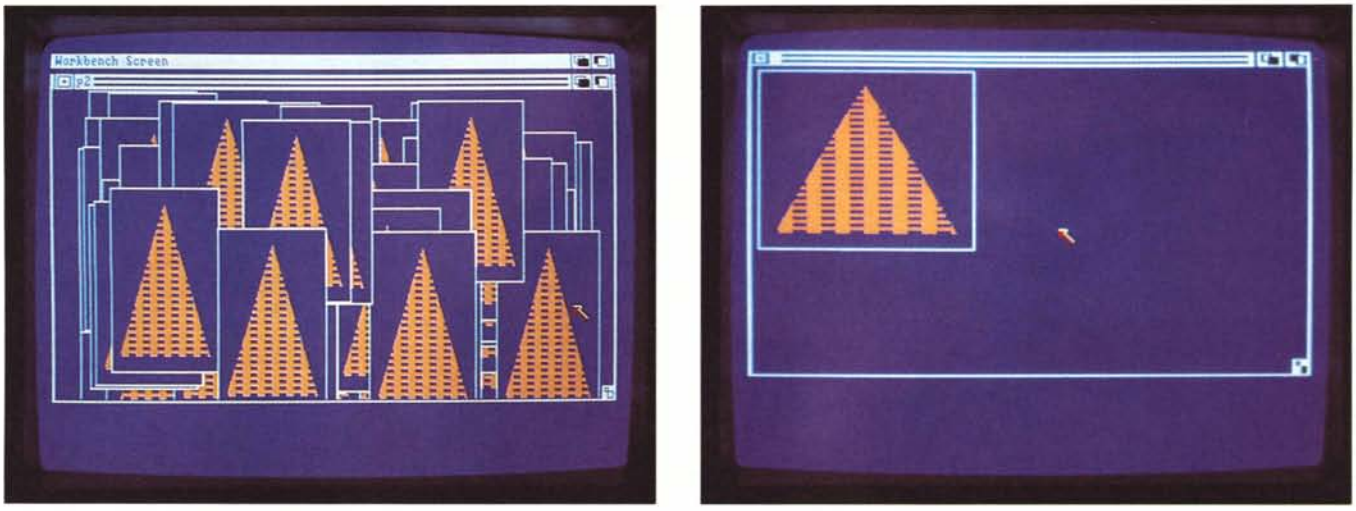
```
WINDOW id
```

per attivarla e basta:

```
WINDOW OUTPUT id
```

In tutt'e tre i casi id sta per identificatore: il numerino usato nella definizione (1 è il numero della finestra OUTPUT del Basic).

Per attivare un nuovo schermo, adopereremo il comando SCREEN indi-



Coi comandi GET e PUT è possibile prelevare aree di schermo per riprodurle nello stesso o in altri schermi.

cando un identificatore (serve per riferire a questo le sue finestre), le dimensioni, il numero di colori e il modo di risoluzione. Come esempio, definiamo un nuovo schermo a 32 colori (quindi bassa risoluzione) a formato intero. Daremo come identificatore 2:

```
SCREEN 2,320,200,5,1
```

Il primo parametro è l'identificatore, seguito dalle dimensioni (320 × 200). 5 indica il numero di bit plane utilizzati (cfr. MC 63) e l'ultimo parametro è il modo bassa risoluzione. Per gli altri modi grafici avremo: 2 per l'alta risoluzione non interlacciata, 3 per la bassa risoluzione interlacciata, 4 per l'alta risoluzione interlacciata. Per chiudere uno schermo liberando così memoria ma perdendo naturalmente anche le finestre in esso contenute, useremo la chiave CLOSE. Volendo chiudere lo schermo appena attivato scriveremo:

```
SCREEN CLOSE 2
```

...ma non lo chiudiamo. Piuttosto, apriamo una bella finestra sul nuovo schermo: basterà indicare 2 (id. di schermo) come ultimo parametro del comando WINDOW:

```
WINDOW 5, «pippo»,(10,10)-(200,100),15,2
```

Per finire ricordiamo che per scrivere o disegnare in una finestra occorre dapprima attivarla (WINDOW id oppure WINDOW OUTPUT id) e resterà tale fino a nuovo ordine.

Colori e Palette

Dicevamo che ogni schermo di Amiga dispone di una propria Palette di colori, comune a tutte le sue finestre, la cui dimensione dipende dal numero di bit-plane utilizzati. Lo schermo standard (quello del Workbench) è formato da due bit-plane quindi i colori disponibili saranno in tutto 4. Per l'esattezza numerati da 0 a 3. Ma come detto eventuali nuovi schermi definiti col comando SCREEN potranno disporre di più bit-plane quindi di un maggiore numero di colori.

Potremo allora variare la palette relativa allo schermo in quel momento attivo, col comando PALETTE nella sintassi:

```
PALETTE idcol, R,G,B
```

idcol è il numero del colore (nello schermo standard da 0 a 3, 0 è il colore di fondo), i valori R, G e B sono le componenti cromatiche di rosso, verde e blu. Immetteremo valori tra 0 e 1, ad esempio:

```
PALETTE 0,1,1,1
```

imposta il colore di fondo bianco,

```
PALETTE 1,0,0,0
```

imposta il colore caratteri nero (ricordiamo che l'assenza di componenti cromatiche dà, ovviamente, il nero mentre la presenza di tutt'e tre, il bianco). Se desideriamo ad esempio un grigio-azzurrino potremo selezionare le componenti così:

```
PALETTE 1,0.5,0.5,0.7
```

Una volta definita la palette (o preso atto del fatto che ne esiste già una di default) potremo selezionare i colori per scrivere o tracciare grafici col comando COLOR:

```
COLOR p,f
```

dove p è il colore della «penna» (numero d'ordine nella palette) e f è il colore del fondo per i caratteri.

L'acchiappapezzetti

Da AmigaBasic è possibile prelevare porzioni di schermo, salvandole momentaneamente in un array, per riprodurle in qualsiasi altra zona dello schermo. Ovvero in qualsiasi altra finestra anche in uno schermo diverso (dunque con risoluzione diversa). Useremo il comando GET per «catturare» un rettangolo di immagine, attireremo la finestra che ci interessa, «scaricheremo» col comando PUT. La fase centrale sarà, ovviamente, omessa se siamo interessati a manipolazioni nella stessa finestra. Procediamo con ordine. Prima di «catturare» occorre dimensionare un array sufficientemente grande da contenere il nostro oggetto. Come da manuale, la formula per ottenere il minimo numero di elementi interi da dimensionare è:

```
6+(Y2-Y1+1)*INT((X2-X1+16)/16)*D
```

dove (X1,Y1) e (X2,Y2) sono gli estremi del rettangolo da salvare, D il numero di bit-plane con cui è formato lo schermo in questione. Facciamo un esempio: preleviamo il rettangolo compreso tra (0,0) e (150,50), dallo schermo standard che è formato da due bit-plane. Valutando la formula di cui sopra con tali valori otteniamo 1026 (spero di non aver sbagliato i calcoli...) quindi la prima operazione sarà:

```
DIM A%(1026)
```

il nome dell'array può essere qualsiasi. Segue la vera e propria operazione di cattura:

```
GET (0,0)-(150,50),A%
```

Esempi di applicazioni dei comandi grafici dell'AmigaBasic.

```
SCREEN 3,640,400,4,4
WINDOW 4,"pippo",,,3
WHILE 1
  x1=RND(1)*640
  x2=RND(1)*640
  y1=RND(1)*400
  y2=RND(1)*400
  LINE (x1,y1)-(x2,y2),k:k=(k+1) MOD 16
  p=p+1
WEND
```

Listato 1

```
WINDOW 2,"pippo"
DIM p%(300)
WHILE p<500
  x1=RND(1)*640
  x2=RND(1)*640
  y1=RND(1)*200
  y2=RND(1)*200
  LINE (x1,y1)-(x2,y2),k:k=(k+1) MOD 4
  p=p+1
WEND
WHILE 1
  GET (0,186)-(617,186),p%
  SCROLL (0,0)-(617,186),0,1
  PUT (0,0),p%
WEND
```

Listato 2

```
DIM a%(3),b%(2000)
a%(0)=&HFF0
a%(1)=&HFFFF
a%(2)=&HFFFF
a%(3)=&HFF0
PATTERN 255,a%
AREA (100,10)
AREA (50, 100)
AREA (150,100)
COLOR 3
AREAFILL
COLOR 2
LINE (40,0)-(160,110),,b
GET (40,0)-(160,110),b%
WHILE 1
  x=RND(1)*500
  y=RND(1)*100
  PUT (x,y),b%,PSET
WEND
```

Listato 3

A questo punto, in A% sarà stivata la nostra porzione di schermo. L'operazione inversa a PUT, con sintassi altrettanto semplice:

PUT c,ar,az

dove c sono le coordinate (assolute o relative col prefisso STEP) del punto di inizio scaricamento, ar l'array in cui è salvata l'immagine, az l'azione da compiere: PSET, PRESET, AND, OR, XOR, PSET accende i punti corrispondenti, PRESET fa l'operazione inversa, AND, OR, XOR le normali operazioni logiche corrispondenti: XOR è l'azione di default. Ad esempio, prendiamo la figura recentemente salvata e scarichiamola con modalità OR a partire dalla coordinata (100,100):

PUT (100,100),A%,OR

già fatto!?!?

Scrolling fine e «grosso»

Col comando SCROLL è possibile far «scrollare» aree di schermo.

La sintassi è molto semplice: basta indicare il rettangolo interessato e lo spostamento orizzontale, verticale o entrambi.

SCROLL (10,10)-(100,100),1,1

effettuerà lo scrolling «diagonale» di un pixel verso destra e uno verso sinistra.

Per spostamenti contrari sarà sufficiente indicare valori negativi.

Se si desidera lo scroll in un solo verso, sarà sufficiente indicare 0 per lo spostamento al quale non siamo interessati.

Combinando il comando SCROLL col GET e PUT appena visti, sarà possibile effettuare scrolling circolari di aree di schermo: ciò che vedremo scomparire da un lato lo vedremo ricomparire dall'altro. È molto semplice: una volta ricavate le coordinate dell'area interessata, prima di effettuare lo scroll salveremo col GET la linea che dovrà scomparire per rimetterla dall'altro lato dopo il comando di scorrimento. Il listato 2 è un esempio di tale procedimento. Ricordiamo che lo stesso trucco era stato utilizzato anche dal lettore Dante Sbrega autore del programma F15 pubblicato sul numero 63.

Per finire

Le ultime tre istruzioni grafiche che vedremo riguardano il riempimento di aree poligonali con possibilità di definirne il pattern. Come noto il pattern è la forma del «pennello» con cui riempire aree. Potremo ad esempio avere aree a pallini, a rete, a quadri, a goccia... secondo il proprio grado di Kitch-eria (!).

Per definire un poligono da riempire si utilizza il comando AREA tante volte quanti sono i vertici (...quindi minimo 3). Una volta definito il poligono, AREAFILL lo riempirà. Se ad esempio vogliamo disegnare un triangolo scriveremo:

```
AREA (100,10)
AREA (50,100)
AREA (150,100)
AREAFILL
```

I primi tre comandi, come detto, servono per definire le coordinate dei vertici. Come al solito queste potranno essere assolute o relative (STEP). Per quanto riguarda AREAFILL, potremo indicare il modo di riempimento 0 (default) o 1. Il modo 1 inverte i colori dell'area selezionata, invece che effettuare il riempimento normale.

Se il pattern «tutto pieno» non è di nostro gradimento, col comando PATTERN potremo cambiarlo, inserendo il nuovo disegno in un qualsiasi array intero (16 bit) formato da un numero di elementi potenza di 2 (1,2,4,8,16...). Sempre col medesimo comando potremo definire il pattern delle linee, semplicemente indicandolo come un valore compreso tra 0 e 65535. Tale dato, visto come numero binario, sarà il pattern di linea. La sintassi è:

PATTERN n,ar

dove n è il pattern delle linee, e ar è l'array che contiene il pattern di riempimento. Come dire: buon divertimento!