SetColor

di Oscar Sillani - Montecorvino Pugliano (SA)

Si sa, nessuno è perfetto, e neanche l'AmigaBasic lo è: escludendo la velocità, vero tallone d'Achille di questo linguaggio (in confronto alle effettive possibilità della macchina), vi sono, qua e là, piccole mancanze, per porre rimedio alle quali sarebbe occorsa pochissima fatica da parte di «mamma Microsoft». Ma a tutto c'è rimedio, e non mi esagera dicendo che è possibile praticamente tutto da Basic, utilizzando appieno le feature di cui tale linguaggio dispone. Per il fedele lettore di questa rubrica che starà pensando «... rieccoci: si parlerà per l'ennesima volta delle funzioni Library...», mi spiace, ma proseguendo nella lettura di queste righe avrà una regolare smentita: spesso inutilizzata, quasi sempre sottoulizzata, stavolta la «protagonista» è la tecnica di programmazione per «sub-programs» (sotto-programmi), caratteristica dell'AmigaBasic a cui si può tranquillamente assegnare il primo premio in efficienza e duttilità: qualcosa di simile alle funzioni Library, ma in Basic e, in più, col vantaggio del «do it yourself», quindi un guadagno in trasparenza d'uso e comprensibilità. Un sub-program fa uso di variabili locali, indipendenti (salvo diversa specifica) dalle variabili del programma principale di cui è ospite; questo vuol dire che, conoscendone i parametri e la sintassi, ogni sub-program può essere «importato» in qualsiasi programma senza bisogno di alcuna modifica per quest'ultimo. Non è esagerato dire che scrivendo un sub-program si aggiunge una nuova istruzione al Basic: ne è prova non solo la trasportabilità di cui sopra, ma anche la distinzione, praticamente nulla, a livello di listato, tra i comandi standard, built-in, del Basic e le istruzioni-sub-program. È quindi possibile costruire «librerie» di sub-program, per gestire problemi e routine ricorrenti o anche per rimediare alle mancanze del Basic. Come unione di queste due finalità, il sub-program SetColor svolge la funzione di editor di colori-schermo a livello Basic. Praticamente in tutti i programmi che fanno uso della grafica è presente un analogo dispositivo per il settaggio/variazione dei colori, purtroppo l'AmigaBasic ne è sprovvisto e chi programma con questo linguaggio ne avrà certamente sentito la mancanza. Sì, è sempre possibile ridefinire i colori da Preferences, ma ciò vale solo se si fa uso dello schermo di default del Basic, cioè lo schermo Workbench. I risultati migliori, in Basic, si ottengono, però, con la definizione di schermi custom con maggiore definizione e/o con più colori. A questo livello non è più possibile agire da Preferences, per cui l'unica alternativa sarebbe arrivare per tentativi ai risultati desiderati. Il sub-program SetColor, fornisce al Basic l'intuitività d'uso di un color-editor, aggiungendo la capacità di adattarsi a qualsiasi combinazione di risoluzione/numero colori e di generare dati direttamente utilizzabili nei programmi Basic. Per attivare SetColor, sia in modo comandi (dalla Output window) sia nel corso di un programma, possono essere usati, indifferentemente, due tipi di sintassi, e cioè:

CALL SetColor (win,scrn, file$)

oppure, più semplicemente:

SetColor win, scrn, file$

Il significato dei parametri passati al sub-program è il seguente:

win: numero della window di cui si vogliono variare i colori; in effetti, si cambieranno i colori all'intero schermo di appartenenza, ma ciò è dovuto al fatto che il Basic seleziona i vari schermi solo tramite le window richiamate dall'utente;

scrn: schermo a cui appartiene la window (1-4: schermi custom/-1: schermo Workbench);

file$: stringa contenente il nome del file che conterrà le linee DATA relative ai dati-palette.

I tre parametri possono essere espressi sia come valori immediati, sia come variabili. Nessun parametro può essere omesso; soltanto nel caso non si voglia ottenere alcun file-dati, il parametro file$ potrà essere costituito da una stringa nulla (" "). L'uso dell'editor è molto intuitivo: basta selezionare col mouse, nella window «SetColor», il colore da modificare, quindi, agendo sulle barre contrassegnate R(ed), G(reen) e B(lue), si possono modificare i valori delle tre componenti luminose. Clickando sulla scritta «Restore» si riportano tutti i colori ai loro valori originali. Agendo, infine, sul close-box di SetColor si esce dall'editor e, nel caso il parametro file$ non sia nullo, i dati relativi ai colori ottenuti vengono scritti nel file specificato, sotto forma di linee DATA (tre valori in virgola mobile, per ogni colore, con range 1.00 - 0.00, indicanti le percentuali decimali delle componenti R-G-B). Il file può esser aggiunto in coda a qualsiasi programma Basic con l'istruzione:

MERGE file$

Un modo molto più veloce di gestire i dati creati con SetColor è «far credere» al Basic che si tratti di dati da poco «copiati» con i comandi «Copy» o «Cut» del menu «Edit». Tali funzioni mantengono i loro dati nel Ram Disk, in un file ASCII di nome «Basic Clip». Sarà quindi sufficiente che il parametro file$ sia costituito dalla stringa «RAM:BasicClip», per far sì che alla semplice attivazione della funzione «Paste» (sempre dal menu «Edit») corrisponda l'inserimento dei dati-palette nel punto del listato Basic identificato dalla posizione del cursore (window «List»).

Ad utilizzare le linee DATA così aggiunte sarà una procedura del tipo:

```
FOR col=0 to ncol
  READ red,green,blue
  PALETTE col,red,green,blue
NEXT col
```

Ovviamente prima di fare ciò, bisogna

```
' Programma dimostrativo
' per l'uso di SetColor
'
SCREEN 1,320,256,5,1
WINDOW 2,"Color test",,23,1
FOR t=6.28 TO 0 STEP -.2
  COLOR (31/6.28)*t
  x1=(150/6.28)*t:y1=(120/6.28)*t
  x=COS(t)*x1:y=SIN(t)*y1
  x1=SIN(t)*x1:y1=COS(t)*y1
  AREA (150+x,120+y)
  AREA (150-x1,120+y1)
  AREA (150-x,120-y)
  AREA (150+x1,120-y1)
  AREA (150+x,120+y)
  AREAFILL
NEXT
SetColor 2,1,"ram:color-data"
WINDOW CLOSE 2:SCREEN CLOSE 1
MERGE "ram:color-data"
END
'
' Inserire qui (con MERGE "SetColor")
' il sub-program per la regolazione
' dei colori.
```

Listato 2

aver già definito uno schermo con capacità di colori adeguata ed avervi aperto una window qualsiasi. La variabile ncol (numero di colori) deve essere corrispondente al numero di colori referenziati nelle linee DATA.

È possibile cambiare a piacere le dimensioni della window «SetColor» regolando le variabili XW e YW (normalmente poste a 13 e 11) all'inizio del sub-program: queste indicano, in caratteri (8x8 pixel), la larghezza e l'altezza della window.

Dimensioni differenti di quest'ultima possono essere utili in presenza di risoluzioni diverse da quella del Workbench. Si sarebbe potuto ricorrere anche alle possibilità di resizing proprie delle window di Amiga, ma ciò avrebbe richiesto maggior memoria (la memoria allocata per una window con resizing è quella relativa alle massime dimensioni raggiungibili), oltre ad appesantire l'algoritmo dell'interfaccia utente. Per utilizzare il sub-program SetColor è importante che questo sia salvato in formato ASCII con:

SAVE "SetColor",a

Questo perché l'istruzione Basic MERGE può leggere solo file del tipo ASCII e non del tipo compattato che l'AmigaBasic utilizza normalmente.

Un esempio di utilizzazione può esser fornito dal secondo listato, un piccolo programma Basic il quale crea uno schermo in bassa risoluzione con 32 colori, vi disegna dei motivi geometrici colorati, quindi richiama SetColor (che sarà stato aggiunto in coda al programma con un comando MERGE): ci si può a questo punto esercitare ad ottenere i risultati cromatici desiderati. Al termine SetColor creerà, nel Ram Disk, il file «ram:color-data», quindi unirà quest'ultimo al programma (per verificare ciò basta osservare il listato per notare l'aggiunta delle linee DATA).

Una curiosità: visto che gli sprite hardware condividono gli ultimi 16 registri di colore-schermo, è possibile cambiare da Basic i colori del puntatore del mouse; questo è, infatti, lo sprite#0 e i registri colore in questione sono il 17, il 18 e il 19.

Anche se lo schermo selezionato non ne fa uso, il contenuto dei suddetti registri può esser variato a piacere tramite una o più istruzioni PALETTE, per ottenere, così, un puntatore «dipinto di nuovo». MC

```
' Utility SETCOLOR --- (c) 1989 Oscar Sillani
' Sintassi:  SetColor w,s,f$
'         o:  Call SetColor (w,s,f$)
' dove  w = numero window
'       s = numero screen
'      f$ = nome output-file
'
  SUB SetColor (wi%,sc%,file$) STATIC
  xw = 13: yw = 11:  'dimensioni window
  WINDOW wi%: colors = WINDOW(6)
  IF colors <> precol THEN
    DIM c%(colors,2),m%(colors,2)
    precol = colors
  END IF
  xd% = xw * 8: yd% = yw * 8:  'dimensioni in pixel
  xb% = xd% / 3: yb% = (yd% - 42) / 16
  c = 0: p = 1 / 16: d = xd% / (colors + 1)
  ScrnP& = PEEKL(WINDOW(7)+46)+44
  ClorMap& = PEEKL(ScrnP&+4)
  ClorTable& = PEEKL(ClorMap&+4)
  FOR k=0 TO colors
    'separzione componenti RGB
    col$ = RIGHT$("00"+HEX$(PEEKW(ClorTable&+ k*2)),3)
    FOR t = 0 TO 2
      c%(k,t) = VAL("&H"+MID$(col$,t+1,1))
      m%(k,t) = c%(k,t)
    NEXT
  NEXT
  WINDOW 18,"SetColor",(0,0)-(xd%-1,yd%-1),26,sc%
  FOR t = 0 TO colors
    LINE (d*t,yd%-16)-STEP(d,15),t,bf
  NEXT:
  LOCATE yw-4
  PRINT PTAB(xb%/2-4)"R"; PTAB(xb%/2+xb%-4)"G"; PTAB(xb%*2+xb%/2-4)"B"
  COLOR 0,1: LOCATE yw-3
  PRINT SPACE$(xw/2-4);"Restore";SPACE$(xw-POS(0)+1): COLOR 1,0
ind:  'selezione colori
  LOCATE yw-2: PRINT PTAB(d/2+d*c-4)"V"
  LINE (0,0)-(xd%,yd%-41),0,bf
  FOR t = 0 TO 2
    LINE (xb% * t,(15-c%(c,t))*yb%)-STEP(xb%-1,yb%-1),1,bf
  NEXT

minp:  'gestione mouse
  WHILE MOUSE(0)=0
    IF WINDOW(6)=0 THEN file: 'la window e' stata chiusa
  WEND
  x = MOUSE(1): y = MOUSE(2): IF POINT(x,y) < 0 THEN minp
  IF y >= yd%-16 THEN
    LOCATE yw-2: PRINT PTAB(d/2+d*c-4)" "
    c = INT(x/d): GOTO ind
  END IF
  IF y >= yd%-32 AND y < yd%-24 THEN
    'funzione Restore
    FOR k = 0 TO colors
      FOR t = 0 TO 2
        c%(k,t) = m%(k,t)
      NEXT
      PALETTE k, c%(k,0)*p, c%(k,1)*p, c%(k,2)*p
    NEXT: GOTO ind
  END IF
  IF y > yd%-42 OR POINT(x,y)=0 THEN minp
  xt = INT(x/xb%)
move:  'simulazione gadget proporzionali
  IF MOUSE(0) > -1 THEN minp
  y = INT(MOUSE(2)/yb%):
  IF y > 15  OR  15-y = c%(c,xt) THEN move
  LINE (xb%*xt,(15-c%(c,xt))*yb%)-STEP(xb%-1,yb%-1),0,bf
  LINE (xb%*xt,y*yb%)-STEP(xb%-1,yb%-1),1,bf
  c%(c,xt) = 15-y
  PALETTE c, c%(c,0)*p, c%(c,1)*p, c%(c,2)*p
  GOTO move
file:  'uscita dal sub-program ed eventuale scrittura file
  WINDOW CLOSE 18: WINDOW 1
  IF file$ <> "" THEN
    OPEN file$ FOR OUTPUT AS 1
    FOR t = 0 TO colors
      PRINT#1,"data ";
      FOR e = 0 TO 2
        PRINT#1,USING "#.###";c%(t,e)*p;
        IF e < 2 THEN PRINT#1,",";
      NEXT: PRINT#1,""
    NEXT: CLOSE 1
  END IF
  END SUB
```

Listato 1