Installazione di una stampante

di Sergio Polini

Abbiamo esaminato le tecniche per controllare una stampante in una applicazione Windows, constatando i vantaggi della indipendenza dall'hardware, cioè della possibilità di inviare ad una stampante comandi che avranno effetto quale che essa sia, senza doverci curare delle idiosincrasie di ognuna. Ci siamo proposti di ottenere un'analoga flessibilità in programmi realizzati con il Turbo Pascal e il suo Turbo Vision; a questo scopo, abbiamo realizzato una unit PTSETUP per l'impostazione delle caratteristiche «temporanee» di una stampante (quali il formato della carta volta per volta utilizzata), e abbiamo iniziato ad esaminare una unit PRDEF per la definizione delle caratteristiche «permanenti» (quali le sequenze di controllo per il grassetto e il sottolineato). Ora vedremo un programma PRINST che consente di installare una stampante per l'uso da parte delle nostre applicazioni Turbo Vision

La volta scorsa ci siamo soffermati sulla interfaccia della unit PRDEF, dedicata alla definizione delle caratteristiche permanenti di una stampante: nome della stampante e del file contenente la sua descrizione, sequenze di controllo per l'inizializzazione, il grassetto e il sottolineato, comando per il movimento per punti e sua risoluzione. Ricordo che per «movimento per punti» intendiamo la possibilità di muovere la testina della stampante in senso orizzontale con una risoluzione maggiore di quanto sarebbe possibile inviando dei normali caratteri; in una stampante Epson ad aghi, ad esempio, possiamo usare il suo modo grafico per disegnare invisibili figure rappresentate da byte nulli, dove ogni byte corrisponde ad una «colonna» di aghi della testina e, quindi, ad un «punto»: inviando un numero di byte inferiore al numero dei punti che esprimono la larghezza di un carattere, la testina si muoverà meno di quanto farebbe se inviassimo uno spazio (ASCII 32).

L'interfaccia della unit PRDEF comprende la dichiarazione di una classe *TPrDefDialog*, una classe derivata da *TDialog* per la realizzazione di dialog box attraverso cui l'utente possa specificare le caratteristiche della stampante che voglia installare sul suo sistema, sia le caratteristiche permanenti, sia quelle temporanee di default; la dialog box, infatti, comprende un pulsante Imposta attraverso il quale si accede ad una dialog box dedicata alle caratteristiche temporanee (tipi carattere disponibili, formato della carta usata, porta cui la stampante è collegata), realizzata mediante la classe *TPrSetupDialog*, dichiarata nella unit PRSETUP.

Abbiamo già visto come è realizzato il constructor della classe *TPrDefDialog*; per prima cosa, vedremo ora il resto della implementazione della unit PRDEF.

Dinamica del dialogo

La classe *TPrDefDialog* comprende, tra le altre, le variabili d'istanza PS e *SetupDone*. Quanto alla prima, il metodo *SetData* le assegna il valore del campo *Setup* del record *TPrinterDef* passato come parametro, mentre il metodo *GetData* effettua l'operazione inversa. Ricordo che, come avevamo visto nel numero di luglio/agosto, quel record comprende i campi per la definizione delle caratteristiche permanenti di una stampante, oltre a quel campo *Setup*, di tipo *PrinterSetup*, dedicato alle caratteristiche temporanee. Queste ultime vengono, per così dire, isolate dalle altre, anche perché vengono impostate mediante un'apposita dialog box.

Quando viene premuto il pulsante *Imposta*, infatti, viene generato un comando *cmSetup*; il metodo *HandleEvent*, non appena ne registra la presenza, attiva una dialog box di tipo *TPrSetupDialog*, dopo averla inizializzata con le informazioni eventualmente contenute in quel campo *Setup* e, quindi, nella variabile *PS*.

Nonostante abbiamo ben distinto tra la definizione delle caratteristiche permanenti e l'impostazione di quelle temporanee, abbiamo anche chiarito che, quando si provvede alle prime, è comunque necessario precisare valori di default per le seconde. Al fine di verificare che l'utente non abbia trascurato tale aspetto, viene assegnato alla variabile *SetupDone* il valore TRUE o FALSE secondo che l'utente abbia chiuso la dialog box di impostazione dopo aver completato il dialogo (premendo il tasto di Invio o il pulsante *Ok* e superando i controlli di validità visti il mese scorso) o rinunciando (con il tasto Esc o il pulsante *Annulla*).

Il metodo *Valid*, quindi, verifica che si siano immessi i nomi della stampante e del suo file di descrizione, che quest'ultimo sia accettabile come nome di file (l'estensione .PRN non va indicata in quanto viene aggiunta automaticamente), che i metodi *Valid* dei controlli ritornino tutti TRUE (a ciò si provvede chiamando *TDialog.Valid*), ma anche che

```
procedure TPrDefDialog.SetData(var Rec);
begin
  TDialog.SetData(Rec);
  PS := TPrinterDef(Rec).Setup;
end;

procedure TPrDefDialog.GetData(var Rec);
begin
  TDialog.GetData(Rec);
  TPrinterDef(Rec).Setup := PS;
end;

procedure TPrDefDialog.HandleEvent(var Event: TEvent);
var
  R: TRect;
  D: PDialog;
begin
  TDialog.HandleEvent(Event);
  if (Event.What = evCommand) and (Event.Command = cmSetup) then begin
    R.Assign(1,2,79,22);
    D := New(PPrSetupDialog, Init(R));
    D^.SetData(PS);
    if DeskTop^.ExecView(D) = cmOk then begin
      D^.GetData(PS);
      SetupDone := True;
    end
    else
      SetupDone := False;
    Dispose(D, Done);
  end;
  ClearEvent(Event);
end;

function TPrDefDialog.Valid(Command: Word): Boolean;
const
  LegalChars: set of Char = ['0'..'9','A'..'Z','a'..'z'];
var
  i: Integer;
  Ok: Boolean;
begin
  Ok := True;
  if Command = cmCancel then begin
    if PS.FontList <> nil then
      Dispose(PS.FontList, Done);
  end
  else if Command <> cmValid then begin
    if Name^.Data^[0] = #0 then begin
      MessageBox(#3'Indicare il nome della stampante', nil,
        mfError or mfOkButton);
      Name^.Select;
      Ok := False;
    end

    else begin
      for i := 1 to Length(FName^.Data^) do
        if not (FName^.Data^[i] in LegalChars) then
          Ok := False;
      if (Length(FName^.Data^) = 0) or (not Ok) then begin
        MessageBox(#3'Nome del file non valido.', nil,
          mfError or mfOkButton);
        FName^.Select;
        Ok := False;
      end;
    end;
    if Ok then
      Ok := TDialog.Valid(Command);
    if Ok and (not SetupDone) then begin
      MessageBox(#3'Impostazione non definita', nil,
        mfError or mfOkButton);
      Ok := False;
    end;
  end;
  Valid := Ok;
end;

end.
```

Figura 1 - La seconda parte della implementazione della unit PRSETUP (la prima parte e l'interfaccia sono state pubblicate nel numero di settembre).

SetupDone sia TRUE; in caso contrario, viene emesso il messaggio d'errore «Impostazione non definita».

Il programma PRINST

Il programma PRINST (figura 2) è il nostro primo esempio di uso delle unit PRSETUP e PRDEF; soprattutto, si tratta di quel programma di utilità cui abbiamo spesso fatto riferimento, che va usato per installare una nuova stampante o per eventualmente modificare le impostazioni di una installazione già effettuata.

Oltre alla classe *TPrInst*, derivata da *TApplication*, viene dichiarata una classe *TCreateTVPRNDIRDlg*, che viene usata per la prima installazione, nel caso non sia stata già creata una directory per le stampanti.

Ricordiamo che vi sono due alternative possibili. Di norma, i file di definizione delle stampanti vengono tutti tenuti in una directory C:\TVPRNS; è possibile scegliere qualsiasi altra directory, che verrà però riconosciuta dai programmi applicativi solo se il suo nome risulterà assegnato ad una variabile TVPRNDIR dell'environment. Il constructor di *TPrInst* si avvale della funzione *ChangeToTVPRNDIR* (illustrata nel numero di giugno) per verificare l'esistenza di una directory per le stampanti; se non la trova, viene chiesto il nome di una nuova directory con la dialog box riprodotta nella figura 3.

Il metodo *Valid* di *TCreateTVPRNDIRDlg* verifica che l'utente immetta un nome valido, provando a creare la nuova directory con *MkDir*. In caso positivo, la nuova directory viene resa corrente mediante *ChDir*, altrimenti l'utente viene avvertito con un messaggio d'errore. Nel caso non si sia scelto il nome C:\TVPRNS per la nuova directory, si produce un messaggio per ricordare all'utente che questa sarà riconosciuta solo se il suo nome sarà assegnato ad una variabile TVPRNDIR dell'environment e che, quindi, si richiede una modifica del file AUTOEXEC.BAT (figura 4).

Il constructor e il metodo *HandleEvent* della classe non richiedono particolari commenti, se non magari per sottolineare la semplicità con cui, in quest'ultimo, si può ottenere la conversione automatica in lettere maiuscole di tutte le minuscole digitate dall'utente.

Una volta creata la directory, potrà essere scelta l'opzione *Nuovo* del menu

```
program PrInst;
(*$X+*)
uses
  Objects, Views, Drivers, Dialogs, Menus, App, MsgBox, StdDlg,
  Dos, PrSetup, PrDef;

const
  cmNewFile = 100;
  cmOpenFile = 101;

type

  TPrInst = object(TApplication)
    PD: TPrinterDef;
    FileName: FNameStr;
    constructor Init;
    destructor Done; virtual;
    procedure InitMenuBar; virtual;
    procedure InitStatusLine; virtual;
    procedure HandleEvent(var Event: TEvent); virtual;
    procedure DoDialog;
    procedure FileNew;
    procedure FileOpen;
  end;

  PCreateTVPRNDIRDlg = ^TCreateTVPRNDIRDlg;
  TCreateTVPRNDIRDlg = object(TDialog)
    Dir: PInputLine;
    constructor Init(Bounds: TRect);
    procedure HandleEvent(var Event: TEvent); virtual;
    function Valid(Command: Word): Boolean; virtual;
  end;

constructor TCreateTVPRNDIRDlg.Init(Bounds: TRect);
var
  R: TRect;
begin
  TDialog.Init(Bounds, 'Crea Directory');
  R.Assign(2,2,48,3);
  Insert(New(PStaticText, Init(R,
    'Non trovata alcuna directory per le stampanti.')));
  R.Assign(2,3,50,4);
  Insert(New(PStaticText, Init(R,
    'Immetti il nome COMPLETO ([drive]:\[percorso])')));
  R.Assign(2,4,50,5);
  Insert(New(PStaticText, Init(R,
    'di una nuova directory:')));
  R.Assign(2,6,50,7);
  Dir := New(PInputLine, Init(R, 67));
  Insert(Dir);
  R.Assign(10,8,22,10);
  Insert(New(PButton, Init(R, 'O~k~', cmOk, bfDefault)));
  R.Assign(30,8,42,10);
  Insert(New(PButton, Init(R, 'Annulla', cmCancel, bfNormal)));
  SelectNext(False);
end;

procedure TCreateTVPRNDIRDlg.HandleEvent(var Event: TEvent);
begin
  if Event.What = evKeyDown then
    Event.CharCode := UpCase(Event.CharCode);
  TDialog.HandleEvent(Event);
end;

function TCreateTVPRNDIRDlg.Valid(Command: Word): Boolean;
var
  Ok: Boolean;
begin
  if (Command = cmCancel) or (Command = cmValid) then
    Ok := True
  else begin
    if Dir^.Data^ <> '' then begin
      Ok := Dir^.Data^[2] = ':';
      if Ok then begin
        (*$I-*) MkDir(Dir^.Data^); (*$I+*)
        Ok := IOResult = 0;
      end
    end
    else
      Ok := False;
    if Ok then begin
      ChDir(Dir^.Data^);
      if Dir^.Data^ <> 'C:\TVPRNS' then
        MessageBox('Verificare che il file AUTOEXEC.BAT'+
          ' contenga una riga:'#13+
          'SET TVPRNDIR='+Dir^.Data^, nil, mfWarning or mfOkButton);
    end
    else begin
      Dir^.Select;
      MessageBox(#3'Nome non valido.', nil, mfError or mfOkButton);
    end;
  end;
  Valid := Ok;
end;

constructor TPrInst.Init;
var
  R: TRect;
  D: PDialog;
begin
  TApplication.Init;
  if not ChangeToTVPRNDIR then begin
    R.Assign(1,1,53,12);
    D := PDialog(ValidView(New(PCreateTVPRNDIRDlg, Init(R))));
    if D <> nil then begin
      if DeskTop^.ExecView(D) = cmCancel then begin
        Done;
        Halt;
      end;
      Dispose(D,Done);
    end;
  end;
end;

destructor TPrInst.Done;
begin
  TApplication.Done;
  RestoreDirectory;
end;

procedure TPrInst.InitMenuBar;
var
  R: TRect;
begin
  GetExtent(R);
  R.B.Y := R.A.Y + 1;
  MenuBar := New(PMenuBar, Init(R, NewMenu(
    NewSubMenu('~P-ile', hcNoContext, NewMenu(
      NewItem('~N-uovo', '', 0, cmNewFile, hcNoContext,
      NewItem('~A-pri...', '', 0, cmOpenFile, hcNoContext,
      NewLine(
      NewItem('~E-sci', 'Alt-X', kbAltX, cmQuit, hcNoContext,
      nil))))),
    nil))));
end;

procedure TPrInst.InitStatusLine;
var
  R: TRect;
begin
  GetExtent(R);
  R.A.Y := R.B.Y - 1;
  StatusLine := New(PStatusLine, Init(R,
    NewStatusDef(0, $FFFF,
      NewStatusKey('~Alt-X~ Esci', kbAltX, cmQuit,
      NewStatusKey('', kbF10, cmMenu,
      nil)),
  nil)));
end;

procedure TPrInst.HandleEvent(var Event: TEvent);
begin
  TApplication.HandleEvent(Event);
  if (Event.What = evCommand) then begin
    case Event.Command of
      cmNewFile: FileNew;
      cmOpenFile: FileOpen;
    else
      Exit;
    end;
    ClearEvent(Event);
  end;
end;

procedure TPrInst.DoDialog;
var
  R: TRect;
  D: PPrDefDialog;
begin
  R.Assign(1,1,79,21);
  D := New(PPrDefDialog, Init(R));
  D^.SetData(PD);
  if DeskTop^.ExecView(D) = cmOk then begin
    D^.GetData(PD);
    if not WritePrinterInfo(FileName, PD) then
      MessageBox('Errore scrittura '+Filename, nil,
        mfError or mfOkButton);
  end;
  Dispose(D, Done);
end;

procedure TPrInst.FileNew;
begin
  FillChar(PD, SizeOf(PD), 0);
  PD.Cmd := HMove[1];
  PD.Setup.OutPort := Ports[1];
  DoDialog;
end;

procedure TPrInst.FileOpen;
var
  P: PView;
begin
  P := ValidView(New(PFileDialog, Init('*.PRN', 'Apri File',
    '~N~ome', fdOpenButton, 100)));
  if P <> nil then begin
    if DeskTop^.ExecView(P) <> cmCancel then begin
      P^.GetData(FileName);
      Dispose(P, Done);
      if ReadPrinterInfo(FileName, PD) then
        DoDialog
      else
        MessageBox('Errore lettura '+Filename, nil,
          mfError or mfOkButton);
    end
    else
      Dispose(P, Done);
  end;
end;

var
  Inst: TPrInst;

begin
  RegisterType(RCollection);
  RegisterType(RFont);
  Inst.Init;
  Inst.Run;
  Inst.Done;
end.
```

Figura 2 - Il programma PRINST, da utilizzare per l'installazione delle stampanti che si intende utilizzare.

File; il metodo *TPrInst.FileNew* provvede ad inizializzare la variabile d'istanza *PD*, di tipo *TPrinterDef*, con valori tutti nulli. Fanno eccezione le stringhe che descrivono la modalità per il movimento per punti e la porta cui la stampante è connessa; come valori di default vengono impostati il modo Epson e la prima parallela (LPT1).

Il metodo *FileNew* potrà essere utilizzato per installare stampanti anche in un secondo momento, quando la directory designata conterrà già uno o più file di descrizione. Il metodo *FileOpen*, dal canto suo, verrà attivato ogni volta che l'utente, scegliendo l'opzione *Apri* del menu *File*, vorrà rileggere ed eventualmente modificare le caratteristiche di una stampante già installata. In questo caso, verrà mostrata la dialog box per la scelta di un file contenuta nella unit STDDLG, la quale proporrà l'elenco dei file .PRN contenuti nella directory delle stampanti. Scelto un file, questo verrà letto mediante la funzione *ReadPrinterInfo* (illustrata nel numero di settembre); le relative informazioni verranno così assegnate ai diversi campi della variabile PD.

Completati i preliminari, sia *FileNew* che *FileOpen* chiamano il metodo *DoDialog*; questo apre una dialog box di tipo *TPrDefDialog* per proporre all'utente l'immissione o la modifica delle caratteristiche della stampante e, se l'utente conferma premendo Invio o il pulsante *Ok*, le informazioni vengono salvate su file mediante la funzione *WritePrinterInfo* (anch'essa illustrata nel numero di settembre).

Le operazioni di lettura e scrittura dei file di descrizione delle stampanti avvengono tutte sulla directory individuata o creata dal programma PRINST all'inizio; sia la funzione *ChangeToTVPRNDIR*, infatti, che il metodo *Valid* di *TCreateTVPRNDIRDlg* rendono corrente la directory destinata a contenere quei file. Quando si esce dal programma, tuttavia, si ritorna automaticamente alla directory in cui si era posizionati quando si era eseguito PRINST; si dovrebbe mantenere PRINST.EXE, infatti, in una directory compresa tra quelle assegnate alla variabile PATH dell'environment, in modo da poter eseguire il programma ovunque si sia posizionati, ad esempio nella subdirectory dedicata ad un programma applicativo Turbo Vision che produrrà elaborati che si vuole stampare con la stampante appena acquistata; è certamente più comodo ritrovarsi, dopo l'installazione, nella directory dell'applicativo piuttosto che in una popolata solo dai file di descrizione delle stampanti. Per ottenere ciò, il destructor di *TPrInst*, dopo aver chiamato *TApplication.Done*, chiama la procedura *RestoreDirectory*; come avevamo visto a giugno, la funzione *ChangeToTVPRNDIR*, prima di cambiare directory, salva il nome di quella corrente in una variabile *OldPath* nascosta nella implementazione della unit PRSETUP e quindi, poiché la funzione viene comunque chiamata (anche se ritorna FALSE e si deve ricorrere a *TCreateTVPRNDIRDlg*), *RestoreDirectory* è in grado in ogni caso di riportarci lì dove eravamo mediante un semplice *ChDir(OldPath)*.

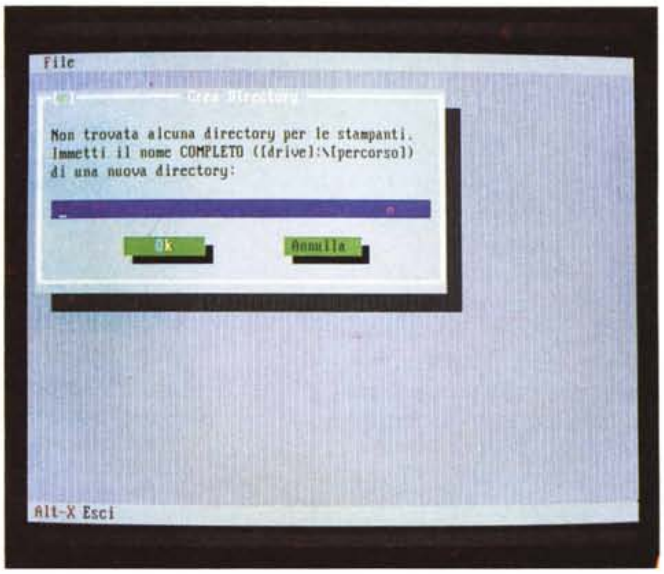

Figura 3 - Se PRINST non trova una directory già predisposta per contenere i file di descrizione delle stampanti, chiede il nome di una nuova directory, che poi provvede a creare.

Figura 4 - Se l'utente indica un nome diverso da C:\TVPRNS per la directory delle stampanti, un messaggio gli ricorda di modificare il file AUTOEXEC.BAT per assegnare quel nome alla variabile TVPRNDIR dell'environment.

Il mese prossimo vedremo come utilizzare nei nostri programmi le informazioni contenute nei file di descrizione delle stampanti; vedremo anche come e dove vi sarà possibile estendere le funzionalità di tutto l'impianto, in modo da ottenere nelle stampe quella maggiore flessibilità (uso di caratteri proporzionali, interlinea variabile, ecc.) cui abbiamo dovuto rinunciare, per motivi di spazio, sulle pagine della rivista.

MC

Sergio Polini è raggiungibile tramite MC-link alla casella MC1166 e tramite Internet all'indirizzo MC1166@mc-link.it.