

# Usare il T<sub>E</sub>X

Alcuni aspetti (non troppo) tecnici della creazione di documenti T<sub>E</sub>X.

## Elaborazione dei documenti T<sub>E</sub>X

Dopo aver presentato [alcune delle potenzialità tipografiche del T<sub>E</sub>X](#), vediamo alcuni dettagli (non troppo) tecnici sull'uso del programma.

Cominciamo con il presentare una sessione tipica di utilizzo del T<sub>E</sub>X.

L'utente lancia il suo editor di testo preferito (un qualsiasi editor va bene, purché permetta di salvare ciò che si scrive in formato testo puro), e scrive il testo che intende scrivere; il testo può essere inserito senza preoccuparsi di quando si va a capo, o di lasciare troppi spazi tra le parole; l'unica accortezza è quella di separare i capoversi con delle linee vuote.

Quando si finisce di scrivere, si salva il documento (in formato testo), e si lancia l'eseguibile del T<sub>E</sub>X dandogli come argomento il documento salvato (comunemente detto *il sorgente*). T<sub>E</sub>X provvederà ad elaborare il testo, formatterà opportunamente i capoversi (riconosciuti dalle linee vuote), sillabando le parole se necessario, comporrà le pagine e salverà il risultato in un file con estensione `dvi`<sup>1</sup> (*DeVice Independent*, indipendente dall'utensile di stampa).

---

<sup>1</sup> a meno che non si usi il PDF<sub>T<sub>E</sub>X</sub> in modalità PDF, nel qual caso si otterrà un file PDF.

Con il file DVI si possono fare tante cose:

- visualizzarlo (e/o stamparlo), usando un opportuno visualizzatore (ogni distribuzione ne fornisce almeno uno);
- convertirlo in POSTSCRIPT, per poterlo visualizzare/stampare/manipolare con appositi programmi, come GHOSTSCRIPT;
- convertirlo in PDF;

eccetera eccetera.

Se si riscontrano errori nel risultato, occorre caricare nuovamente l'editor, riaprire il sorgente e modificarlo, salvarlo e ricompilarlo: questo è il famigerato *ciclo visualizza/modifica/compila* (*view/edit/compile cycle*), ed è una delle due cose che rende il T<sub>E</sub>X fastidioso persino per coloro che sono entusiasti del programma.

Per ridurre al minimo questo fastidio gli utenti T<sub>E</sub>X hanno sviluppato una convenzione a cui aderiscono praticamente tutti i visualizzatori e molti editor; questa convenzione permette di passare dalla visualizzazione alla modifica (e viceversa) con fatica ridotta: se si riscontra durante la visualizzazione un punto in cui si desidera apportare una modifica, si può fare doppio click nel punto della modifica, ed il visualizzatore provvederà a lanciare l'editor desiderato in un punto molto vicino a quello desiderato<sup>2</sup>.

Rimane ancora un problema: se il sorgente che si scrive è un semplice file di testo, come è possibile inserire titoli, figure, tabelle, formule eccetera? La risposta

---

<sup>2</sup> molto vicino, ma non esatto, poiché una precisione assoluta richiederebbe una quantità spropositata di informazioni, rendendo la elaborazione e visualizzazione del DVI molto lenta.

è semplice: tutto ciò che non è semplice testo viene creato con appositi comandi; i comandi vengono inseriti direttamente nel testo, e non si ha pertanto una immediata visione di come appariranno le cose: occorre compilare il sorgente e visualizzare il risultato per conoscere i risultati.

Ciò è platealmente diverso dall’approccio visuale a cui ci hanno abituato i *word processor*, ma è ben poco prezzo da pagare visto la qualità di ciò che si ottiene.

Ovviamente questo approccio testuale favorisce la creazione di documenti ben strutturati, e non a caso i formati tendono a favorire una separazione tra struttura ed aspetto: l’utente specifica cosa è ogni parte di testo (capitoli, paragrafi, indici, le varie parti delle voci bibliografiche), e definisce *separatamente* il loro aspetto, cosa che tra l’altro garantisce coerenza all’interno del documento, e spesso una certa uniformità di aspetto tra i documenti: una volta che si è trovata una buona formattazione per le varie parti dei documenti, è possibile riutilizzarla senza difficoltà in ogni documento; ovviamente, se si desidera un aspetto diverso, è possibile crearne di nuovi, ed è possibile mantenere *librerie* di stili (con terminologia L<sup>A</sup>T<sub>E</sub>X; in C<sup>O</sup>N<sup>T</sup>E<sup>X</sup>T vengono chiamati *environments*, ovvero *ambienti*).

Lavorare con documenti ben strutturati è certamente vantaggioso, quando si opera con programmi come il T<sub>E</sub>X (ed è comunque una buona abitudine da prendere anche con i *word processor*: imparate a conoscere e ad usare gli stili—o i corrispettivi strumenti del vostro *word processor*). Ma il T<sub>E</sub>X permette anche di lavorare di “fantasia”, nel senso che, invece di comandi di alto livello come **titolo**, **capitolo** o **appendice**, è possibile utilizzare i comandi interni del T<sub>E</sub>X specifici per la formattazione (selezione manuale dei font, spostamenti ben precisi all’interno della pagina, eccetera).

Ovviamente, questo tipo di manipolazione non solo non è incoraggiata (e dovrebbe comunque essere riservata a pagine particolari come le copertine), ma è anche difficoltosa, poiché per poter sapere ciò che si ottiene (ed apportare eventuali correzioni) occorre eseguire il famigerato ciclo.

I risultati ottenibili sono comunque di qualità superiore (ricordate che state lavorando con un sistema tipografico), è possibile riciclarli senza fatica per nuovi documenti o nuove situazioni, e non c'è rischio di perdere informazioni<sup>3</sup>.

Ancora una volta, messi vantaggi e svantaggi su una bilancia, oserei dire che il piatto dei vantaggi sia più ricco di quello degli svantaggi.

## T<sub>E</sub>X ed i font

Un altro aspetto che l'utente *word processor* troverà oltremodo complesso da usare in un sistema tipografico come il T<sub>E</sub>X è la gestione dei font. Poiché nel T<sub>E</sub>X la gestione dei font è effettivamente alquanto articolata, mi limiterò ad accennare gli aspetti fondamentali.

I sistemi tipografici come il T<sub>E</sub>X non si preoccupano realmente dell'aspetto dei caratteri; gli unici dettagli importanti sono le dimensioni dei singoli caratteri, le coppie di accostamento (*kerning pairs*)<sup>4</sup> e la presenza di eventuali legature. L'insieme di queste informazioni è in genere indicato con il termine 'metriche'.

---

<sup>3</sup> cosa che può accadere con quei *word processor* in cui non è possibile vedere i codici: non vi è mai capitato di posizionare un riquadro un po' fuori dalla pagina, senza più riuscire a renderlo visibile?

<sup>4</sup> ad esempio, quando una V ed una A maiuscole sono vicine, come in VA, è meglio avvicinarle l'una all'altra, come in VA; l'entità dell'avvicinamento cambia da font a font, ed in base alle lettere.

Per usare un font in T<sub>E</sub>X è quindi necessario che il T<sub>E</sub>X abbia accesso alle metriche del font. A seconda di come il font viene distribuito, queste informazioni possono essere direttamente accessibili, oppure da porre in un formato intellegibile al T<sub>E</sub>X.

Un'altro aspetto del rapporto tra T<sub>E</sub>X e font è che il T<sub>E</sub>X non favorisce l'uso di un numero eccessivo di font (e varianti come corsivo, grassetto, maiuscoletto, eccetera) diversi (ed è bene che faccia così, perché un eccessivo uso di font e varianti è una cattiva abitudine tipografica). Ma potete fidarvi: quasi tutte le limitazioni imposte dal T<sub>E</sub>X servono a scoraggiare abusi tipografici, e sono aggirabile nel caso in cui gli abusi li si voglia realmente fare.

Infine, un altro problema legato all'uso dei font con T<sub>E</sub>X non dipende realmente dal T<sub>E</sub>X stesso, ma da idiosincrasie dei programmi di visualizzazione. Esso ha a che fare con la resa su schermo scadente di alcuni documenti prodotti dal T<sub>E</sub>X.

Il programma specifico per generare font per T<sub>E</sub>X (il METAFONT) crea i font come bitmap (non vettoriali). Questo implica, tra le altre cose, che un font creato col METAFONT per una certa risoluzione/dimensione non può essere usato a risoluzioni/dimensioni diverse, pena la perdita di qualità; occorre quindi generare una copia del font per ogni risoluzione/dimensione voluta.

I motivi per cui i font vengono creati in questo modo sono vari. Innanzi tutto, quando il T<sub>E</sub>X fu creato non erano ancora diffusi computer abbastanza potenti da permettere un uso indiscriminato dei font vettoriali. Ma vi sono anche motivi tecnici per cui i font vengano creati così: si ha infatti un controllo punto per punto dei caratteri, se si desidera, ed alcuni effetti come la riduzione di spessore nei punti di intersezione delle linee sono notevolmente semplificati.

Uno dei maggiori motivi è comunque nel fatto che Knuth fa parte della vecchia scuola tipografica che ritiene che ogni dimensione dei caratteri debba avere caratteristiche diverse: in piccole dimensioni il rapporto fra altezza e larghezza dei caratteri tende ad essere piccolo (dando una forma all'incirca quadrata), mentre a dimensioni maggiori i caratteri hanno generalmente una forma più allungata.

Infine, diverse risoluzioni degli strumenti di stampa implicano una diversa quantità di dettagli (un numero eccessivo di dettagli su strumenti a bassa risoluzione ‘sporca’ il carattere, un numero ridotto di dettagli su strumenti ad alta risoluzione riduce il gusto del carattere); piuttosto che cercare un compromesso, Knuth ha allora deciso di generare immagini diverse per diverse risoluzioni.

Quali sono allora le implicazioni dell'uso di font bitmap? Cosa c'è di sbagliato in essi?

In linea di principio, nulla; e la qualità di stampa, anziché risentirne, è certamente migliore (l'immagine di ogni carattere è infatti disegnata specificatamente per una certa coppia dimensione/risoluzione).

Il problema si ha con la diffusione di documenti elettronici. Infatti, il formato consigliabile per la circolazione di documenti elettronici è il PDF, ma purtroppo il lettore di file PDF più diffuso (Acrobat Reader) è scarsamente dotato in termini di rappresentazione delle immagini (e dei font) bitmap: il che vuol dire che la visione tramite Acrobat Reader di un documento PDF scritto con font bitmap è a dir poco fastidiosa; d'altra parte la stampa di questi stessi documenti su strumenti che usano la stessa risoluzione del font risulta invece eccellente (i font hanno in genere risoluzioni di 300 o 600 dpi, ed ormai tutte le stampanti non hanno problemi ad usare queste risoluzioni).

L'unico modo di aggirare questa limitazione<sup>5</sup> dell'Acrobat Reader è usare font vettoriali.

Il font predefinito del T<sub>E</sub>X (un font disegnato dallo stesso Knuth), il *Computer Modern*, esiste in versione vettoriale gratuita; questo font non contiene però lettere accentate (che vengono create sovrapponendo l'accento alla lettera non accentata), e pertanto la sillabazione delle parole accentate non è sempre ottimale; esistono estensioni del Computer Modern (la più usata è il font *European Computer Modern*) che contengono lettere accentate, ma le versioni vettoriali di questi font sono a pagamento<sup>6</sup>.

Si possono però usare con il T<sub>E</sub>X font diversi dal *Computer Modern* e dalle sue estensioni; e se si ha già una versione vettoriale dei font che si desidera usare, l'unica cosa che occorre creare affinché il T<sub>E</sub>X li possa usare sono le metriche del font. Per di più, esistono già (e sono generalmente distribuite con il T<sub>E</sub>X) le metriche dei font più diffusi (come il Times, l'Arial, il Courier eccetera).

---

<sup>5</sup> in realtà Acrobat Reader ha molte altre limitazioni, e vari problemi, che hanno portato un 'pilastro' della comunità T<sub>E</sub>X internazionale come Donald Arseneau ad affermare che “‘Acrobat’ si chiama così per le contorsioni a cui ci si deve sottoporre per aggirarne gli errori” (*post* su [news://comp.text.tex](http://comp.text.tex) del 16 maggio 2001). Maggiori informazioni sui problemi di Acrobat che interessano la comunità T<sub>E</sub>X possono essere trovate all'indirizzo <http://www.micropress-inc.com/acr5bugs.htm>.

<sup>6</sup> in realtà un recente progetto denominato *TeXtrace* è riuscito a creare font vettoriali per lo *European Computer Modern*; possiamo quindi aspettarci tra breve di averle disponibili gratuitamente

## T<sub>E</sub>X e le immagini

In “[Scrivere, un’arte ... grafica?](#)” (paragrafo “... and friends”, [pagina 8](#)) ho detto una mezza bugia a proposito della gestione delle immagini in T<sub>E</sub>X: ho detto che il T<sub>E</sub>X poteva inserire qualsiasi immagine nel vostro documento.

Il problema è in realtà un po’ più complesso; il T<sub>E</sub>X infatti non inserisce mai alcuna immagine nel documento: si limita a riservare lo spazio necessario all’inclusione, inclusione che poi avverrà per opera del visualizzatore (nel caso di visione del DVI su schermo), del convertitore (nel caso di conversione in POSTSCRIPT, PDF, HTML etc), del programma atto alla stampa (nel caso si stampi direttamente il DVI con un opportuno programma).

L’inclusione di un’immagine in un documento T<sub>E</sub>X richiede quindi due cose:

1. che il T<sub>E</sub>X conosca le dimensioni dell’immagine;
2. che il visualizzatore/convertitore/elaboratore del DVI sappia come gestire il formato richiesto.

In linea di massima, il [problema 1](#) non è un vero problema, in quanto è sempre possibile specificare manualmente le dimensioni dell’immagine; per alleggerire il lavoro dell’utente, alcuni formati T<sub>E</sub>X come L<sup>A</sup>T<sub>E</sub>X e C<sup>O</sup>N<sup>T</sup>E<sup>X</sup>T hanno strumenti che permettono al T<sub>E</sub>X di determinare automaticamente le dimensioni di certi tipi di immagini.

Il [problema 2](#) è ovviamente legato al visualizzatore/convertitore/elaboratore in uso, e dipende quindi o dalla distribuzione che si usa (nel caso di visualizzazione/stampa del DVI), oppure dal formato in cui si intende convertire il documento T<sub>E</sub>X.

Poiché i formati di conversione più ambiti sono il POSTSCRIPT, il PDF e l'HTML, vedremo una breve panoramica per ogni formato.

- Immagini per POSTSCRIPT

Ovviamente, il formato immagini più congeniale ai documenti POSTSCRIPT è il formato EPS, che è un formato vettoriale; per le immagini bitmap, è da suggerire il formato JPEG; la conversione da JPEG in EPS consiste infatti nell'aggiunta di qualche linea di codice POSTSCRIPT che permettono all'interprete POSTSCRIPT di leggere direttamente il file JPEG.

- Immagini per PDF

Poiché il POSTSCRIPT (e l'EPS) è un linguaggio di programmazione, mentre il PDF è semplicemente un linguaggio descrittivo, i visualizzatori PDF non sono forniti delle capacità necessarie alla elaborazione dei file EPS. La conseguenza immediata di ciò è che non è possibile usare immagini EPS direttamente nei documenti PDF; tuttavia, PDF e POSTSCRIPT sono abbastanza simili da permettere la conversione del formato EPS in PDF (esistono in giro vari tools per la conversione; vedi anche [“Oltre al T<sub>E</sub>X”](#) in [“Come e dove procurarsi il T<sub>E</sub>X”](#)).

D'altro canto, con PDF vi sono meno problemi per la gestione delle immagini bitmap: è possibile infatti inserire immagini JPEG e PNG direttamente, ed esistono strumenti che permettono la conversione in JPEG e PNG degli altri formati bitmap.

- Immagini per HTML

Nel caso dei documenti HTML, la capacità di visualizzare o meno un certo formato di immagine dipende esclusivamente dal browser. Vi sono però dei formati universalmente accettati, come il JPEG ed il GIF; i browser più recenti accettano anche il formato PNG (che è un formato recente).

Sia JPEG che PNG sono formati compressi, in modo che le immagini occupino meno spazio che se fossero descritte punto per punto. Vediamo ora le principali differenze tra i due formati.

- ▷ JPEG

Il formato JPEG è stato creato per la compressione delle immagini “continue” (come ad esempio le foto); questo ha come principale conseguenza che le immagini più semplici hanno una scarsa qualità (nelle zone a tinta unica vengono introdotti degli artefatti non presenti nell’immagine originale).

Un altro svantaggio del JPEG è che il suo formato di compressione è *lossy*, ovvero a perdita d’informazione: per aumentare la compressione vengono eliminate alcune informazioni, con conseguente perdita di dettagli; è però possibile decidere, al momento della creazione del JPEG, quanta perdita si può accettare.

Il grosso vantaggio del JPEG è il ridottissimo ingombro (ed è possibile ridurre ulteriormente pur di rinunciare a più dettagli).

▷ PNG

Il formato PNG è stato creato come alternativa gratuita e libera da brevetti del più diffuso formato GIF; sono inoltre state aumentate le caratteristiche del formato (rispetto al predecessore GIF) ed è stata migliorata la compressione.

La compressione nei PNG è *lossless*, ovvero senza perdita d'informazioni; questo implica da un lato una fedelissima riproduzione, dall'altra un maggior ingombro in termini di spazio.

La scelta del formato è quindi una funzione dello spazio che ci si vuole permettere e del tipo/qualità di immagini che si desidera inserire; come linee guida generali, il formato JPEG (a bassa compressione/alta qualità se si desidera mantenere i dettagli) è da preferire per foto ed immagini simili; per gli altri usi è consigliabile il PNG.

## T<sub>E</sub>X visuale?

Per coloro che non possono fare a meno dell'approccio visuale (e non solo per loro: una comodità è una comodità), sono disponibili in circolazione sia dei T<sub>E</sub>X visuali (commerciali), sia *emulatori* per certuni formati del T<sub>E</sub>X; il più diffuso tra gli ultimi è certamente LyX, un emulatore grafico per L<sup>A</sup>T<sub>E</sub>X, disponibile per sistemi operativi POSIX<sup>7</sup>, ma eseguibile anche sotto Windows, a condizione di avere un server X.

---

<sup>7</sup> UNIX, LINUX, le varie varianti BSD e simili

## Colofon

Queste pagine sono state composte dal `CONTEXT`; tutti gli errori di battitura ed eventuali errori di altro genere sono da ascrivere all'autore dei fogli, e non al programma di composizione.

***bilotta78@hotpop.com***