Claudio Beccari Tommaso Gordini

Introduzione alle codifiche in entrata e in uscita

26 ottobre 2017

GUIT

Associati anche tu al GUIT

Fai click per associarti

L'associazione per la diffusione di TEX in Italia, riconosciuta ufficialmente in ambito internazionale, si sostiene *unicamente* con le quote sociali.

Se anche tu trovi che questa guida tematica gratuita ti sia stata utile, il mezzo principale per ringraziare gli autori è diventare socio.

Divenendo soci si ricevono gratuitamente:

- l'abbonamento alla rivista ArsTEXnica;
- il DVD TEX Collection;
- un eventuale oggetto legato alle attività del GUIT.

L'adesione al GUIT prevede un quota associativa compresa tra 12,00 € e 70,00 € a seconda della tipologia di adesione prescelta ed ha validità per l'anno solare in corso.

Indice

Introduzione **ii**

1 Codifica in entrata e codifica in uscita **1**
1.1 Introduzione 1
1.2 Le tre distinte codifiche di TEX 3
1.2.1 La codifica di ingresso 11
1.2.2 La codifica di uscita 14

2 Specificare la codifica giusta **18**
2.1 Scoprire la codifica di input 20
2.2 Cambiamento della codifica 21
2.3 Collage di contributi diversi 25

3 Considerazioni riassuntive **26**

Introduzione

Il problema delle codifiche è delicato e spesso difficile da capire da parte di chi non conosce il funzionamento interno del proprio calcolatore.

Le richieste di aiuto sul forum GuIT sono incessanti; dipendono da tanti fattori fra i quali quello dell'aggiornamento del software, il cambio di calcolatore, il passaggio a un sistema operativo diverso. Ma la domanda ricorrente è sempre quella: "Ma prima funzionava tutto correttamente; perché ora le lettere accentate non vengono più composte?" Oppure: "Non riesco a capire: ora le lettere accentate sono sostituite da rombi neri con un punto interrogativo al centro; che cosa succede?"

La questione delle codifiche è un terreno scivoloso; spero di poterlo rendere meno infido con questa breve guida tematica, tratta in sostanza per buona parte da una appendice della cosiddetta GuidaGuIT, (Beccari, 2012).

Chi vuol arrivare subito alle conclusioni operative salti direttamente all'ultimo capitolo di questa guida: 3. Poi se vuole capire perché vengono suggerite quelle soluzioni può tornare qui all'inizio per vedere le cose tecniche che portano a quelle conclusioni.

Capitolo 1

Codifica in entrata e codifica in uscita

1.1 Introduzione

L'argomento della codifica è uno di quelli che più travagliano gli utenti del sistema TEX. Per cercare di fare un po' di luce su questi argomenti, bisogna rifarsi al calcolatore che elabora i dati. Al suo interno ogni tipo di informazione è "scritto" in termini di cifre binarie '0' e '1'. Sia ben chiaro: queste cifre non sono davvero scritte nel modo che noi umani comunemente associamo al verbo 'scrivere'. I segni '0' e '1' sono essi stessi dei modi codificati per descrivere gli stati di certe quantità fisiche all'interno del calcolatore: saranno tensioni oppure correnti positive o negative; saranno delle piccolissime parti di materiale magnetico magnetizzato con il polo nord in alto oppure in basso; saranno delle cariche elettriche positive o negative localizzate in certi cristalli di silicio o di altro materiale semiconduttore; in realtà non ci interessa andare così a fondo nella struttura della macchina per capire che le cifre binarie '0' e '1' sono modi convenzionali per descrivere uno o l'altro dei due stati opposti che possono assumere quelle grandezze fisiche in quei determinati punti della macchina. A noi interessa il significato che di volta in volta viene attribuito a quei modi di essere di quelle grandezze fisiche.

I segnali che vengono trasmessi da un punto all'altro della macchina sono sequenze di 'zeri' e 'uni' e l'informazione che essi trasportano da un punto all'altro della macchina dipende dal trasmettitore e dal ricevitore che usano un codice per attribuire questo significato e per comportarsi di conseguenza. Ogni 'zero' e ogni 'uno' di una sequenza si chiama *bit*, un breve acronimo diventato nome comune, ottenuto da "Binary unIT", unità binaria, e rappresenta la minima quantità di informazione che può essere contenuta in un messaggio.

A livello macroscopico, quando noi battiamo un tasto sulla tastiera questa invia una sequenza di bit particolare (ovviamente diversa da tasto a tasto) all'unità centrale di elaborazione che ne farà l'uso corrispondente al ruolo che in quel momento la tastiera sta svolgendo – la tastiera non svolge solo il ruolo di strumento di scrittura. La tastiera

fisica non manda all'unità centrale una lettera, ma una sequenza di bit; otto bit (oggi[1]) formano un *byte*; un byte di otto bit può assumere 256 configurazioni diverse: 00000000, 00000001, 00000010, 00000011, ..., 11111111, associabili, come numerazione binaria ai numeri decimali 0, 1, 2, 3, ..., 255. Ma, ecco, già questa è una codifica (anche se non è l'unica lettura possibile), perché a ogni sequenza di bit viene associata una lettura in termini di numero binario; insomma con codifiche diverse (modi di interpretazione diversi) lo stesso byte può assumere significati diversi.

Ma è solo una faccia della questione che riguarda la codifica. Torniamo alla nostra tastiera: la sequenza di bit che una data tastiera invia all'unità centrale è la stessa quando si preme lo stesso tasto ma, per esempio, il secondo tasto da sinistra della terza fila dall'alto significa 'w' con una tastiera QWERTY ma significa 'z' con una tastiera QZERTY. Per facilitare la lettura agli umani, sui tasti sono disegnati determinati segni, ma la tastiera fisica è del tutto indifferente a ciò che è disegnato sui tasti. Tra la tastiera e l'unità centrale di elaborazione c'è quindi un 'filtro' che traduce la sequenza di bit inviati dal tasto premuto in un segno piuttosto che nell'altro a seconda di come l'operatore ha configurato la macchina quando ha risposto ad una apposita domanda che gli ha fatto il programma di installazione del sistema operativo: "che tipo di tastiera hai?"; questo è particolarmente evidente quando si installa una distribuzione Linux, perché questo sistema operativo non è legato a nessun particolare tipo di hardware, nel senso che dovrebbe essere in grado di adattarsi a qualsiasi tipo di hardware di qualunque marca e prodotto da qualunque fabbricante.

Il filtro, comunemente chiamato *driver* di tastiera, può però essere cambiato a piacere; l'utente può installare diversi driver di tastiera sulla sua macchina, e li può rendere attivi (uno alla volta, evidentemente) in modo da poter toccare gli stessi tasti, ma inviare il segno 'w' oppure il segno σ, o qualunque altro segno di qualunque alfabeto che il driver faccia corrispondere a quel famoso secondo tasto da sinistra della terza fila dall'alto. Si tratta di codifiche diverse: cambiando driver si cambia il modo di interpretare la stessa sequenza di bit.

Nello stesso modo possiamo distinguere una codifica d'ingresso quando l'unità centrale di elaborazione (CPU, *central processing unti*) e il programma che in quel momento sta "girando" ricevono una sequenza di bit da qualunque periferica capace di inviare alla CPU segnali formati da sequenze di bit, e una codifica di uscita quando sono l'unità centrale di elaborazione e il programma che vi sta girando in quel momento che inviano segnali alle periferiche. I concetti di 'ingesso' e di 'uscita' si riferiscono quindi ai segnali che entrano nella CPU per essere elaborati, oppure che vi escono dopo l'elaborazione per essere utilizzati da "terze parti".

Nel caso particolare del sistema TEX, ci ritroviamo con l'unità centrale sulla quale gira il programma che chiamiamo editor; i segnali di entrata arrivano dalla tastiera attraverso il suo driver o dal disco fisso (o da qualunque altro dispositivo di memorizzazione di massa) attraverso un altro driver specifico. L'unità centrale e l'editor che vi sta girando

[1] In passato il byte era definito come la più piccola sequenza di bit associabile a un indirizzo di memoria del calcolatore; questi "bocconi" di informazione potevano essere formati da 7, o da 8, o anche da 9 bit; potevano anche avere tre bit in più se erano dei byte auto"correttivi; ma questo esula completamente dall'argomento di questo testo. Basti sapere che "oggi" tutti i sistemi di calcolo usano byte di 8 bit.

inviano segnali al video e al disco fisso (o altro dispositivo) ciascuno attraverso il suo specifico driver. L'editor opportunamente azionato può inviare segnali anche al sistema operativo per ordinargli di mettersi ad eseguire uno dei programmi di composizione del sistema TEX; la CPU sospende l'esecuzione dell'editor e lo mette da parte, carica e lancia l'esecuzione di uno dei programmi del sistema TEX, e assieme richiedono al disco fisso di inviare all'unità centrale i segnali corrispondenti al testo da comporre con il mark-up di LATEX; il flusso di segnali viene elaborato dalla CPU e il risultato viene inviato nuovamente al disco fisso e viene chiuso il programma di composizione richiamando dal suo riposo l'editor; la CPU invia all'editor segnali che il processo di composizione è andato a buon fine, e l'editor rilancia segnali per informare l'unità centrale che può attivare il programma di visualizzazione; il gioco ricomincia: l'unità centrale sospende l'editor e lo rimette da parte, apre o riprende il programma di visualizzazione precedentemente messo da parte e gli dice di riprendere il suo lavoro; questo programma a sua volta richiede al disco di lanciargli i segnali dei dati da visualizzare e provvede con suoi driver interni a mandare i debiti segnali al video affinché esso mostri sullo schermo il risultato della composizione; eventualmente l'operatore dà ordine all'editor di mettere in stampa il prodotto della composizione, ma sebbene le macchine periferiche siano diverse, i concetti generali di collaborazione fra le varie parti della macchina, i programmi e le periferiche sono sempre gli stessi.

Non continuo perciò a descrivere questo dialogo fra sistema operativo, CPU, periferiche come la tastiera, il video e i dischi e gli altri sistemi di memorizzazione di massa, la memoria volatile interna, chiamata RAM, per tenere da parte i programmi che in quel momento sono inoperosi, perché il discorso diventerebbe lunghissimo e noiosissimo. Metto in risalto che ogni volta che si ha un dialogo fra una parte e l'altra della macchina o fra un programma e l'altro, i segnali sono in codice e ogni macchina periferica o centrale può decifrarli attraverso gli appositi driver.

Che cosa c'entra allora l'operatore"/compositore in tutta questa storia di codifiche? Perché il sistema operativo non provvede da solo a gestire queste cose senza tirare in ballo quel poveraccio del compositore? Con i programmi e i sistemi chiusi questo in genere succede, per esempio con la componente Writer di LibreOffice questi problemi non si presentano; il sistema TEX non è un sistema chiuso in sé stesso, e non solo perché è un sistema che deve poter lavorare su qualunque tipo di hardware, con qualunque periferica e sotto il controllo di qualunque sistema operativo, ma anche perché vuole poter essere totalmente staccato da simili situazioni contingenti affinché i suoi file siano trasportabili da una macchina all'altra anche con sistemi operativi completamente diversi.

Il sistema TEX al suo interno funziona con ulteriori codifiche sue proprie.

1.2 Le tre distinte codifiche di TEX

Usando i programmi di composizione del sistema TEX siamo spesso chiamati a specificare la codifica di ingresso attraverso il pacchetto inputenc o di uscita attraverso il pacchetto fontenc; il suffisso 'enc' ricorda la parola inglese *encoding*, cioè codifica.

Come dice la parola 'fontenc', questa ha a che fare con i file nei quali sono memorizzati i "disegni" dei caratteri di un dato font, quelli che servono per 'scrivere' il documento nel

32	␣	48	0	64	@	80	P	96	‘	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	’	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	

Tabella 1.1: I 95 caratteri ASCII stampabili. Il carattere ASCII 32 è lo spazio, qui reso visibile (␣) mediante il comando `\textvisiblespace`. Il carattere ASCII 127 (il novantaseiesimo della tabella) non è un carattere stampabile.

file di uscita. D’altro canto si è parlato spesso di codifica dei caratteri con cui si scrive il file *.tex* da comporre e si è usata la parola ‘inputenc’, quindi si è parlato di codifica di entrata. Di mezzo, fra la scrittura del file sorgente e la scrittura del file composto, c’è il programma di composizione che usa una sua codifica interna per rappresentare i vari caratteri; non dobbiamo preoccuparci di quest’ultima codifica, perché le interfacce fra la codifica di entrata, quella interna e quella di uscita sono proprio fornite dai due pacchetti citati, inputenc e fontenec.

Infatti la codifica in entrata, quella che serve per creare o modificare i file *.tex*, riguarda il modo di funzionare dello *shell editor* con il quale si eseguono queste operazioni e con il quale si salva il file *.tex*. La codifica in uscita riguarda le polizze dei caratteri che si usano per la composizione. Fra queste due codifiche si trova il programma di composizione pdflatex che usa una sua codifica interna. Detto in altri termini: l’editor non si preoccupa del fatto che il testo e il mark-up che esso legge dal disco oppure dalla tastiera verrà successivamente elaborato da un programma invece che da un altro; l’importante è che passi le informazioni giuste mediante il file che il successivo programma di composizione leggerà dal disco, in modo che l’elaborazione dell’informazione contenuta nel disco produca la composizione tipografica desiderata. Ricordiamoci anche che lo *shell editor* con il quale creiamo o modifichiamo e salviamo il file *.tex* non fa propriamente parte del sistema TEX; questo interviene solo per trasformare il file sorgente *.tex* nel file composto *.pdf*.

Questa comunicazione fra l’editor e il programma di composizione si ottiene mediante l’invocazione dei pacchetti inputenc e fontenc con le opzioni appropriate.

Quando si invoca il pacchetto inputenc con una qualunque opzione di codifica che il

pacchetto accetti, si stabilisce una corrispondenza fra la codifica dei caratteri immessi con la tastiera nello *shell editor* (che esso trascrive nel file *.tex* e che mostra graficamente sullo schermo), e quanto dovrà fare il successivo programma pdflatex per tradurlo nella sua codifica interna quando, leggendo il file *.tex*, ne eseguirà la composizione.

Quando invece si invoca il pacchetto `fontenc` con una qualunque opzione accettata da questo pacchetto, si stabilisce una corrispondenza fra il codice interno di pdflatex e i vari segni delle polizze dei font usati, caratterizzati dalla codifica specificata e trascritti da pdflatex sia nella pagina in lavorazione, sia nelle pagine emesse nel file di uscita, le pagine composte che finalmente saranno visualizzate sullo schermo o stampate su carta.

Si noti che i codici emessi dal driver della tastiera, o letti dal disco sotto forma di sequenza di bit, possono essere letti anche come numeri binari e come i loro corrispondenti decimali, e noi possiamo identificare il carattere 'w', per esempio, con il numero decimale 119 e possiamo dire; "viene inviato il carattere 119 dalla tastiera all'editor". Il numero decimale corrisponde quindi ad una specie di indirizzo che corrisponde alla posizione della lettera 'w' in una certa tabella di caratteri corrispondente ad una certa codifica.

In sostanza la tastiera invia alla CPU delle sequenze di byte che le indicano quale tasto è stato premuto e se contemporaneamente erano stati premuti uno o più dei tasti di controllo, come il tasto delle maiuscole o quello delle maiuscole permanenti, il tasto etichettato Ctrl, o quello etichettato Alt, eccetera. In base alle impostazioni del sistema il driver della tastiera traduce questa sequenza di byte in un'altra sequenza, spessissimo costituita da un solo byte, che indica l'indirizzo del segno secondo la codifica scelta. Se si tratta di un unico byte il cui primo bit sia 0, allora il carattere corrisponde ad uno di quelli indicati nella tabella 1.1, che è comune a tutte le altre codifiche di entrata.

Se si tratta di un solo byte e il primo bit vale 1, esso può corrispondere ad un carattere diverso a seconda della codifica; nella tabella 1.2 sono riportati i vari caratteri con indirizzi compresi fra 128 e 255 corrispondenti ad alcune delle codifiche più comuni: *ansinew* adatta ai sistemi Windows (praticamente coincide con la *code page* 1252, per la quale si potrebbe specificare l'opzione *cp1252*); *latin1* adatta ai sistemi Linux e Macintosh; *latin9* adatta agli stessi sistemi ma con alcuni caratteri diversi, compreso quello dell'euro; *applemac* adatta ai sistemi Macintosh per i quali si voglia mantenere l'uso della codifica specifica, disponibile con i sistemi operativi precedenti al Mac OS X, ma usabile anche con questo sistema operativo; *cp437* corrisponde alla codifica delle macchine che usavano il sistema operativo DOS; *cp1250* corrisponde alla codifica delle macchine Windows degli anni '90 ed è la versione usata in quei paesi dell'Europa orientale che usano certe varianti dell'alfabeto latino.

Nella tabella alcuni caratteri sembrano non produrre nessun risultato; uno è il carattere relativo allo spazio indivisibile (*no break space*); un altro è il marcatore di parola composta (*compound word marker*) che di fatto, con il sistema TEX, è sostituito dal segno di cesura facoltativa reso con il comando `\-`.

Come si vede non c'è praticamente nessuna differenza, tranne per la colonna intestata `latin9` per soli nove caratteri (il simbolo €, № 164; la Š, № 166; la š, № 168; la Ž, № 180; la ž, № 184; la legatura Œ, № 188; la legatura œ, № 189; la Ÿ, № 190; e il carattere № 255 che non viene usato) e quindi la codifica *latin1* o, se si preferisce, la codifica *latin9*, possono essere usate con tutti e tre i sistemi operativi senza problemi, almeno per quel

che riguarda i caratteri dal 161 al 255 e tranne quei pochi glifi diversi con la codifica *latin9*. Si consiglia la codifica *latin1* anche perché molti dei restanti segni sono poi tutti da rendere in uscita con l'uso del pacchetto textcomp che provvede a estendere il numero di glifi disponibili con un altro centinaio di simboli alfabetici e analfabetici.

Le altre colonne sono invece molto diverse dalla prime tre; lasciando perdere le ultime due colonne (che si riferiscono per lo più a situazioni particolari e che non riguardano la grande maggioranza degli utenti italiani che dispongono di macchine "moderne") la codifica *applemac* è abbastanza diffusa fra gli utenti delle macchine Macintosh. ***Si sconsiglia l'uso della codifica applemac con vigore***. Infatti scambiarsi file codificati in questo modo fra utenti che usano diverse macchine, non tutte Macintosh, vuol dire produrre fastidio e irritazione fra tutti gli autori che collaborano ad un progetto collettivo. La codifica *latin1* va benissimo per tutte le macchine moderne più diffuse; la codifica *utf8*, di cui si parlerà fra poco, va ancora meglio.

Tanto per essere espliciti, la frase:

> La città è più bella, perché così può essere più pulita.

scritta con la codifica *applemac* e aperta con un editor impostato per la codifica *latin1* appare sullo schermo così:

> La citt^ pi bella, perchŽ cos“ pu˜ essere pi pulita.

La codifica UNICODE è la più generale: è quasi universale, nel senso che si riferisce a parecchie migliaia o decine di migliaia di segni, non solo quelli delle codifiche a 256 caratteri, e comprende ogni genere di alfabeto, ogni verso di scrittura, ogni collezione di ideogrammi e di grafemi. La lista dei segni è talmente lunga che per consultarla è meglio riferirsi alle pagine UNICODE reperibili sul sito del consorzio omonimo: http:/"/www.unicode.org"/Public"/UNIDATA"/UnicodeData.txt. I codici UNICODE possono contenere fino a 6 cifre esadecimali (3 byte), di cui la prima può assumere solo i valori 0 e 1; quindi essi vanno da 000000 a 10FFFF (cioè 1 114 112 segni; in teoria potrebbe arrivare fino a 1FFFFF e quindi avrebbe spazio per 2 097 152 differenti segni). La codifica UTF-8 è un modo particolare, un algoritmo, per generare il codice UNICODE di ogni carattere in modo da usare meno byte per indirizzare i segni che occupano le prime posizioni della codifica UNICODE; a tutti gli effetti pratici possiamo identificare la codifica UTF-8 (*utf8* per il pacchetto inputenc) con la codifica UNICODE. Per saperne di più sulla codifica UTF-8 si può vedere il sito http:/"/it.wikipedia.org"/wiki"/UTF-8. Con due byte solamente si possono indirizzare tutti i segni per gli alfabeti latino (anche con diacritici), greco, cirillico, copto, ebraico e arabo; per tutti gli ideogrammi (cinese giapponese e coreano) occorrono tre byte; UTF-8 usa quattro byte solo per segni particolarissimi che vengono usati piuttosto raramente.

In modo analogo la codifica di uscita, quella da specificare al pacchetto fontenc, indica la posizione che un dato carattere occupa nel file che contiene le informazioni per disegnare ogni glifo sullo schermo, o sulla carta, o sulla periferica di uscita; la codifica di uscita, quindi, riguarda i font con cui viene composto il documento finale; non per niente quando viene composto questo capoverso l'informazione che viene memorizzata nel file di uscita,

prima ancora di indicare le posizioni dei caratteri del font da usare, è `\T1/lmr/m/n/10.95`, cioè, in codice, l'agglutinazione delle informazioni relative (*a*) alla codifica del file che contiene le istruzioni per 'disegnare' i singoli caratteri (`T1`); (*b*) alla famiglia di caratteri (`lmr`, Latin Modern Roman); (*c*) alla serie dei caratteri (`m`, serie di media nerezza); (*d*) alla forma dei caratteri (`n`, caratteri normali diritti); (*e*) al corpo (`10.95`, corpo di 10.95 pt, quello che normalmente viene chiamato per brevità corpo di 11 pt).

Va detto che in teoria il pacchetto per la codifica d'ingresso *utf8* potrebbe codificare più di un milione di segni; però la cosa sarebbe poco pratica e probabilmente impossibile da realizzare anche con i calcolatori più moderni che possono usare memorie di grandissime dimensioni. Per cui il pacchetto inputenc con l'opzione *utf8* si limita a definire solo le codifiche dei caratteri che possono venire effettivamente rappresentati con i font di uscita; per poterlo fare deve conoscere le codifiche di tutti i font che verranno usati nel corpo del documento. Per questo sarebbe opportuno caricare il pacchetto inputenc solo dopo aver caricato tutti gli altri pacchetti che si riferiscono ai font. In particolare sarebbe opportuno caricare prima, non solo il pacchetto fontenc con le opzioni che si desiderano, ma anche il pacchetto textcomp e ogni altro pacchetto per l'uso di collezioni di simboli speciali non compresi in quelli rappresentabili con la codifica *T1* né con la codifica *TS1* di textcomp. In pratica il pacchetto inputenc, quando viene specificata l'opzione *utf8*, attiva una serie di comandi da eseguire solo al momento di cominciare la composizione del documento, quando tutti i pacchetti dei font sono già stati caricati; in pratica, quindi, sebbene sia preferibile caricare inputenc dopo gli altri pacchetti relativi ai font di uscita, l'utente non deve preoccuparsene perché ci pensa inputenc stesso a sistemare le cose in modo opportuno.

Tabella 1.2: Alcune codifiche di ingresso per i caratteri latini

Codice	ansinew	latin1	latin9	applemac	cp437	cp1250
128	€			Ä	Ç	€
129				Å	ü	
130	‚			Ç	é	‚
131	*ƒ*			É	â	
132	„			Ñ	ä	„
133	…			Ö	à	…
134	†			Ü	å	†
135	‡			á	ç	‡
136	ˆ			à	ê	
137	‰			â	ë	‰
138	Š			ä	è	Š
139	‹			ã	ï	‹
140	Œ			å	î	Ś
141				ç	ì	Ť

Continua nella pagina seguente

Continua dalla pagina precedente

Codice	ansinew	latin1	latin9	applemac	cp437	cp1250
142	Ž			é	Ä	Ž
143				è	Å	Ź
144				ê	É	
145	‘			ë	æ	‘
146	’			í	Æ	’
147	“			ì	ô	“
148	”			î	ö	”
149	•			ï	ò	•
150	–			ñ	û	–
151	—			ó	ù	—
152	˜			ò	ÿ	
153	™			ô	Ö	™
154	š			ö	Ü	š
155	›			õ	¢	›
156	œ			ú	£	ś
157				ù	¥	ť
158	ž			û	Pts	
159	Ÿ			ü	*f*	ź
160				†	á	
161	¡	¡	¡	°	í	ˇ
162	¢	¢	¢	¢	ó	˘
163	£	£	£	£	ú	Ł
164	¤	¤	€	§	ñ	¤
165	¥	¥	¥	•	Ñ	Ą
166	¦	¦	Š	¶	ª	¦
167	§	§	§	ß	º	§
168	¨	¨	š	®	¿	¨
169	©	©	©	©	¬	©
170	ª	ª	ª	™	½	Ş
171	«	«	«	´	¼	«
172	¬	¬	¬	¨	¡	¬
173				≠	«	
174	®	®	®	Æ	»	®
175	¯	¯	¯	Ø		Ż
176	°	°	°	∞		°
177	±	±	±	±		±
178	²	²	²	≤		˛
179	³	³	³	≥		ł
180	´	´	Ž	¥		´
181	µ	µ	µ	µ		µ

Continua nella pagina seguente

Continua dalla pagina precedente

Codice	ansinew	latin1	latin9	applemac	cp437	cp1250
182	¶	¶	¶	∂		¶
183	·	·	·	∑		·
184			ž	∏		¸
185	¹	¹	¹	π		ą
186	º	º	º	∫		ş
187	»	»	»	ª		»
188	¼	¼	Œ	º		Ľ
189	½	½	œ	Ω		˝
190	¾	¾	Ÿ	æ		ľ
191	¿	¿	¿	ø		ż
192	À	À	À	¿		Ŕ
193	Á	Á	Á	¡		Á
194	Â	Â	Â	¬		Â
195	Ã	Ã	Ã	√		Ă
196	Ä	Ä	Ä	ƒ		Ä
197	Å	Å	Å	≈		Ĺ
198	Æ	Æ	Æ	∆		Ć
199	Ç	Ç	Ç	«		Ç
200	È	È	È	»		Č
201	É	É	É	…		É
202	Ê	Ê	Ê			Ę
203	Ë	Ë	Ë	À		Ë
204	Ì	Ì	Ì	Ã		Ě
205	Í	Í	Í	Õ		Í
206	Î	Î	Î	Œ		Î
207	Ï	Ï	Ï	œ		Ď
208	Ð	Ð	Ð	–		Đ
209	Ñ	Ñ	Ñ	—		Ń
210	Ò	Ò	Ò	“		Ň
211	Ó	Ó	Ó	”		Ó
212	Ô	Ô	Ô	‘		Ô
213	Õ	Õ	Õ	’		Ő
214	Ö	Ö	Ö	÷		Ö
215	×	×	×	◊		×
216	Ø	Ø	Ø	ÿ		Ř
217	Ù	Ù	Ù	Ÿ		Ů
218	Ú	Ú	Ú	⁄		Ú
219	Û	Û	Û	¤		Ű
220	Ü	Ü	Ü	‹		Ü

Continua nella pagina seguente

Continua dalla pagina precedente

Codice	ansinew	latin1	latin9	applemac	cp437	cp1250
221	Ý	Ý	Ý	›		Ý
222	Þ	Þ	Þ	fi		Ţ
223	ß	ß	ß	fl		ß
224	à	à	à	‡	α	ŕ
225	á	á	á	·	β	á
226	â	â	â	‚	Γ	â
227	ã	ã	ã	„	π	ă
228	ä	ä	ä	‰	Σ	ä
229	å	å	å	Â	σ	Ĺ
230	æ	æ	æ	Ê	µ	ć
231	ç	ç	ç	Á	γ	ç
232	è	è	è	Ë	Φ	č
233	é	é	é	È	θ	é
234	ê	ê	ê	Í	Ω	ę
235	ë	ë	ë	Î	δ	ë
236	ì	ì	ì	Ï	∞	ě
237	í	í	í	Ì	ϕ	í
238	î	î	î	Ó	ε	î
239	ï	ï	ï	Ô	$\cap$	ď
240	ð	ð	ð	ﹾ	$\equiv$	đ
241	ñ	ñ	ñ	Ò	$\pm$	ń
242	ò	ò	ò	Ú	$\geq$	ň
243	ó	ó	ó	Û	$\leq$	ó
244	ô	ô	ô	Ù		ô
245	õ	õ	õ	ı		ő
246	ö	ö	ö	ˆ	$\div$	ö
247	÷	÷	÷	˜	$\approx$	÷
248	ø	ø	ø	¯	°	ř
249	ù	ù	ù	˘	·	ů
250	ú	ú	ú	˙	•	ú
251	û	û	û	˚	$\surd$	ű
252	ü	ü	ü	¸	^{n}	ü
253	ý	ý	ý	˝	^{2}	ý
254	þ	þ	þ	˛	■	ţ
255	ÿ	ÿ		ˇ		˙

Bisogna però ricordare che fra l'editor e il programma di composizione non c'è un collegamento diretto, ma i due programmi comunicano solo attraverso il file *.tex*.

Ecco perché questo file dovrebbe poter trasferire l'informazione della codifica di

ingresso, che l'editor ha usato per scriverlo, modificarlo e salvarlo, al programma di composizione affinché questo possa recuperare l'informazione corretta per poter elaborare il testo sorgente in modo da comporre il testo nel file di uscita sotto forma di documento composto in modo tipograficamente corretto.

D'altro canto sarebbe opportuno che quando un file sorgente viene generato o modificato mediante un editor usando una certa codifica, esso venisse salvato su disco con la stessa codifica. Sembra ovvio, ma non è così: dipende dalla qualità dell'editor.

Per esperienza chi scrive può affermare che egli conosce due editor TEXShop (specifico delle macchine Macintosh) e TeXworks (multipiattaforma) che accettano delle righe di configurazione che permettono di adattare le impostazioni del programma alle informazioni fornite con queste righe; se ne parlerà più avanti. C'è anche l'editor Aquamacs (solo per macchine Macintosh) che può fare più o meno le stesse cose, ma con righe di configurazione che seguono una sintassi diversa da quella dei programmi menzionati sopra; anche emacs (multipiattaforma) accetta le stesse righe di configurazione che usa Aquamacs. Esistono altresì gli editor TeXstudio e Texmaker che riescono a distinguere automaticamente se un file sorgente è stato creato o modificato con la codifica UTF-8 oppure con la codifica ISO Latin 1; in realtà sono in grado di distinguere i file codificati UTF-8 dagli altri, per cui un file che non sia UTF-8 viene classificato come ISO Latin 1. Questa capacità in generale è sufficiente, ma la codifica ISO Latin 1 non è la sola codifica dove le lettere accentate nazionali sono codificate con un solo byte ma, come si è visto nella tabella 1.2, molte codifiche diverse hanno i caratteri nazionali codificati con un solo byte, quindi il riconoscimento ISO Latin 1 potrebbe essere sbagliato; non succede spesso perché la codifica *applemac* viene usata relativamente poco, ma questo poco è già troppo... Più avanti si vedrà come riconoscere quale sia la codifica di un file *.tex*.

I dettagli sono molto tecnici e si omettono; non conosco nessun documento dove questi dettagli siano descritti in modo comprensibile; per quel che riguarda la codifica di uscita esiste il file *encguide.pdf*, leggibile con il solito comando `texdoc encguide`, dove vengono descritte alcune delle codifiche latine, in particolare la codifica *T1*.

Però non è particolarmente difficile leggere i file, per esempio, *latin1.def* e *t1enc.def*[2]; il primo descrive la corrispondenza fra i caratteri della tastiera codificati secondo la norma ISO 8859-1, nota anche come ISO Latin 1, e la codifica interna di pdflatex. Il secondo descrive la corrispondenza fra la codifica interna di pdflatex e quella dei font usati per comporre il documento.

1.2.1 La codifica di ingresso

Se si apre il file *latin1.def*, si vede che le prime righe definiscono mediante il comando `\ProvideTextCommandDefault` il significato di default di alcuni comandi interni generalmente conosciuti da parte di pdflatex; per esempio, il primo comando

```
\ProvideTextCommandDefault{\textdegree}{\ensuremath{{^\circ}}}
```

dice che se `\textdegree` non è ancora stato definito, esso va composto di default in modo matematico con un circoletto ad esponente. Naturalmente esistono dei pacchetti per i

[2] Si trovano entrambi nella cartella..."`/tex/latex/base/`.

font di uscita che possono definire o ridefinire questo comando di default per produrre un segno più adeguato, disegnato apposta, ma in mancanza di questi font la definizione di default produce un risultato accettabile.

Infatti `\textdegree`, tramite il pacchetto textcomp, produce il segno °, mentre il circoletto ad esponente è °, che, come si vede, è leggermente più grande ma è accettabilissimo.

Ci sono anche dei comandi del tipo:

```
\ProvideTextCommandDefault{\textcent}
  {\TextSymbolUnavailable\textcent}
```

che emettono un messaggio di errore se si usa il comando `\textcent`; questo succede perché il segno ¢ non è facilmente simulabile in un modo che vada bene per ogni font, codifica, famiglia, serie, forma e corpo; per produrlo ci vuole dunque l'uso di una polizza di caratteri in uscita che contenga questo segno e che perciò dia una definizione valida per `\textcent`.

Dopo altri comandi di questa specie, vengono definite le corrispondenze fra i caratteri con gli indirizzi da 161 a 255 nella codifica ISO 8859-1 e la codifica interna di pdflatex. Per esempio la definizione

```
\DeclareInputText{164}{\textcurrency}
```

specifica che introducendo con la tastiera nello *shell editor* il carattere che il driver della tastiera trasforma nel codice 164, questo produce sullo schermo il segno ¤. È vero che fra i primi comandi si dichiara che il segno corrispondente al comando `\textcurrency` non è disponibile, ma se questo comando è già stato definito o viene ridefinito in altro modo, questo segno potrebbe venire stampato nel modo giusto e non verrebbe emesso nessun avviso di simbolo non disponibile; infatti qualche riga più in alto il segno è stato stampato correttamente, perché la composizione di questo testo si affida anche al pacchetto textcomp che permette di usare i segni della collezione Text Companion Font illustrati nella figura 1.1.

Per i caratteri in entrata non strettamente ASCII, come le lettere nazionali si hanno poi svariate definizioni del tipo seguente, valido per la lettera 'ï' che nella codifica di entrata ISO Latin 1 occupa la posizione 239:

```
\DeclareInputText{239}{\"\i}
```

Questa dichiarazione sostituisce il carattere 'ï' contenuto nel file di entrata con quanto è contenuto nel secondo argomento, ma del quale l'utente normale non deve preoccuparsi; esso serve per porre la dieresi sulla lettera 'ı' senza puntino indicato con la macro `\i`. Oggi l'utente normale non deve preoccuparsi della 'i senza puntino' (a meno che non scriva in irlandese o in turco) perché ci pensa sempre LaTeX a usarla quando occorre, almeno in modalità testo. Si vede dunque che anche in questo caso il carattere non appartenente al sottoinsieme ASCII viene sostituito con codici interni di LaTeX.

Si noti dunque che con la codifica di entrata (le altre disponibili con il sistema TeX sono concepite nello stesso modo ma, ovviamente, con definizioni diverse, come è stato

	'00	'01	'02	'03	'04	'05	'06	'07	'10	'11	'12	'13	'14	'15	'16	'17	
'000	` 0	´ 1	ˆ 2	˜ 3	¨ 4	˝ 5	° 6	ˇ 7	˘ 8	¯ 9	˙ 10	¸ 11	˛ 12	' 13	14	15	"00
'020	16	17	" 18	19	20	— 21	— 22	23	← 24	→ 25	‿ 26	‿ 27	‿ 28	‿ 29	30	31	"10
'040	ƀ 32	33	34	35	$ 36	37	38	' 39	40	41	* 42	43	, 44	= 45	. 46	/ 47	"20
'060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	58	59	⟨ 60	— 61	⟩ 62	63	"30
'100	64	65	66	67	68	69	70	71	72	73	74	75	76	℧ 77	78	◯ 79	"40
'120	80	81	82	83	84	85	86	Ω 87	88	89	90	⟦ 91	92	⟧ 93	↑ 94	↓ 95	"50
'140	` 96	97	⋆ 98	o|o 99	† 100	101	102	103	104	105	106	107	108	⚭ 109	♪ 110	111	"60
'160	112	113	114	115	116	117	118	119	120	121	122	123	124	125	~ 126	= 127	"70
'200	˘ 128	ˇ 129	″ 130	‶ 131	† 132	‡ 133	‖ 134	‰ 135	• 136	°C 137	$ 138	¢ 139	f 140	₵ 141	₩ 142	₦ 143	"80
'220	₲ 144	₱ 145	£ 146	₨ 147	⁇ 148	¡ 149	₫ 150	TM 151	‰ 152	¶ 153	₪ 154	№ 155	⅍ 156	℮ 157	° 158	SM 159	"90
'240	{ 160	} 161	¢ 162	£ 163	¤ 164	¥ 165	¦ 166	§ 167	¨ 168	© 169	ª 170	ↄ 171	¬ 172	℗ 173	® 174	‾ 175	"A0
'260	° 176	± 177	² 178	³ 179	´ 180	µ 181	¶ 182	· 183	※ 184	¹ 185	º 186	√ 187	¼ 188	½ 189	¾ 190	₠ 191	"B0
'300	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	"C0
'320	208	209	210	211	212	213	× 214	215	216	217	218	219	220	221	222	223	"D0
'340	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	"E0
'360	240	241	242	243	244	245	÷ 246	247	248	249	250	251	252	253	254	255	"F0
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,00000	x-height	4,30450 pt
spazio interparola	3,33251 pt	larghezza del quadrato	9,99755 pt
allungamento interparola	1,66625 pt	spazio extra	1,11083 pt
accorciamento interparola	1,11083 pt	corpo nominale	10,00000 pt

Figura 1.1: I font con la codifica *TS1*

precedentemente mostrato con alcuni esempi nella tabella 1.2) ogni tasto che produca nello *shell editor* un segno diverso dai caratteri ASCII viene trasformato in un segno visibile sullo schermo e nel file *.tex* che si sta scrivendo o modificando. Quando il file *.tex* verrà letto da pdflatex ogni carattere che vi si trova viene sostituito da comandi interni di LaTeX e verrà trattato come tale, finché non verrà il momento di trasferire il testo composto nel buffer di uscita costituito dalla famosa scatola 255 riservata da TeX a questo scopo.

Si ricorda che i primi 128 caratteri di ogni codifica di ingresso corrispondono sempre ai caratteri definiti dall'*American Standard Code for Information Interchange* – ASCII, che comprendono i 32 codici per l'azionamento delle telescriventi[3], le cifre, i segni di

[3]Cose del passato, ma non completamente. Infatti contengono anche i codici di fine riga e di tabulazione che il sistema TeX sa trattare in modo particolare; i codici di tabulazione per l'editor serve per far spostare il cursore fino alla successiva posizione di tabulazione, ma pdflatex tratta quel codice come un semplice spazio. Invece i codici di fine riga <LF> e <CR> sono usati variamente dai diversi sistemi operativi e, quindi, dagli editor che lavorano su macchine con sistemi operativi diversi. Anche questo è un tormentone, come le codifiche, ma i programmi del sistema TeX sono già predisposti per evitare all'utente di doversene preoccupare. I vari editor invece possono avere qualche problemino quando viene aperto un file composto con un codice di fine riga diverso da quello che l'editor si aspetta. Di solito gli editor multipiattaforma

interpunzione e l'alfabeto latino di 26 lettere non accentate, in forma maiuscola e minuscola, per un totale di $128 - 32 = 96$ segni stampabili (in realtà il centoventottesimo non è stampabile e quindi ne restano solo 95). Da una codifica ISO all'altra cambiano i secondi 128 caratteri, come si è visto con la tabella 1.2, ed è per questo che esistono diversi nomi per distinguere le varie codifiche di ingresso. Ognuno dei file *.def* relativo alla codifica di ingresso definisce i comandi testuali di default e la corrispondenza fra i codici da 128 a 255 con i comandi interni di LaTeX (in alcune codifiche gli indirizzi da 128 a 160 non ricevono nessuna definizione).

Attenzione: normalmente ogni *shell editor* salva i file con la codifica predefinita nella sua configurazione. TeXShop e TeXworks possono fare di più: possono salvare un file scritto con una certa codifica anche con una codifica diversa da quella corrispondente alla sua configurazione. Di solito questa funzionalità non è un problema, anzi può essere molto utile per cambiare codifica ad un file. Bisogna solo stare attenti a non confondersi sbagliando preferenze nel pannello di dialogo per salvare o esportare un file, oppure nella apposita finestrella in basso a destra nella cornice di TeXworks; non è comune fare questi errori, ma vale la pena segnalare la necessità di stare attenti. Questa funzionalità, per altro, si rivela utilissima per cambiare codifica ad un file, come si vedrà più avanti.

1.2.2 La codifica di uscita

La codifica dei font latini *T1* nelle prime 32 posizioni contiene glifi di servizio che comprendono gli accenti ed altri segni speciali; i successivi 95 caratteri sono i caratteri ASCII; dalla posizione 127 alla posizione 255 la polizza di caratteri contiene quasi tutti i segni necessari per scrivere in tutte le lingue dell'Europa occidentale; per l'Europa orientale alcuni segni vengono creati per sovrapposizione dell'accento alla lettera base, ma sono pochissimi i segni che richiedono questo trattamento; si veda per esempio la tabella 1.2.

Il file *t1enc.def* contiene le definizioni delle corrispondenze fra i comandi interni di LaTeX e le posizioni dei singoli segni nella tabella del font di uscita. La sintassi è un po' diversa da quella usata per la codifica di ingresso, ma il significato è altrettanto comprensibile. Per esempio:

```
\DeclareTextAccent{\`}{T1}{0}
\DeclareTextAccent{\'}{T1}{1}
```

definiscono gli accenti grave e acuto mettendo in relazione i comandi LaTeX \` e \' con la posizione dei corrispondenti segni (0 e 1, rispettivamente) nella polizza conforme alla codifica *T1*. Di fatto questi accenti possono servire solo per essere sovrapposti a lettere che non hanno corrispondenza nella codifica del font di uscita *T1*, per esempio: ȁ.

Per la cediglia, che è un segno da mettere sotto, non sopra una lettera, la definizione è molto più complessa:

```
\DeclareTextCommand{\c}{T1}[1]
  {\leavevmode\setbox\z@\hbox{#1}\ifdim\ht\z@=1ex\accent11 #1%
```

sono configurabili in modo da riconoscere un certo codice di fine riga che userà anche per salvare il file che si sta scrivendo o modificando, ma possono essere configurati per riconoscere in lettura fine riga diversi.

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	` 0	´ 1	ˆ 2	˜ 3	¨ 4	˝ 5	˚ 6	ˇ 7	˘ 8	¯ 9	˙ 10	¸ 11	˛ 12	‚ 13	‹ 14	› 15	"00
’020	“ 16	” 17	„ 18	« 19	» 20	– 21	— 22	23	0 24	1 25	J 26	ff 27	fi 28	fl 29	ffi 30	ffl 31	"10
’040	␣ 32	! 33	" 34	# 35	$ 36	% 37	& 38	’ 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	"20
’060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	< 60	= 61	> 62	? 63	"30
’100	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79	"40
’120	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	\ 92	] 93	ˆ 94	_ 95	"50
’140	‘ 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111	"60
’160	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	{ 123	| 124	} 125	˜ 126	- 127	"70
’200	Ă 128	Ą 129	Ć 130	Č 131	Ď 132	Ě 133	Ę 134	Ğ 135	Ĺ 136	Ľ 137	Ł 138	Ń 139	Ň 140	Ŋ 141	Ő 142	Ŕ 143	"80
’220	Ř 144	Ś 145	Š 146	Ş 147	Ť 148	Ţ 149	Ű 150	Ů 151	Ÿ 152	Ź 153	Ž 154	Ż 155	IJ 156	İ 157	đ 158	§ 159	"90
’240	ă 160	ą 161	ć 162	č 163	ď 164	ě 165	ę 166	ğ 167	í 168	ľ 169	ł 170	ń 171	ň 172	ŋ 173	ő 174	ŕ 175	"A0
’260	ř 176	ś 177	š 178	ş 179	ť 180	ţ 181	ű 182	ů 183	ÿ 184	ź 185	ž 186	ż 187	ij 188	¡ 189	¿ 190	£ 191	"B0
’300	À 192	Á 193	Â 194	Ã 195	Ä 196	Å 197	Æ 198	Ç 199	È 200	É 201	Ê 202	Ë 203	Ì 204	Í 205	Î 206	Ï 207	"C0
’320	Ð 208	Ñ 209	Ò 210	Ó 211	Ô 212	Õ 213	Ö 214	Œ 215	Ø 216	Ù 217	Ú 218	Û 219	Ü 220	Ý 221	Þ 222	SS 223	"D0
’340	à 224	á 225	â 226	ã 227	ä 228	å 229	æ 230	ç 231	è 232	é 233	ê 234	ë 235	ì 236	í 237	î 238	ï 239	"E0
’360	ð 240	ñ 241	ò 242	ó 243	ô 244	õ 245	ö 246	œ 247	ø 248	ù 249	ú 250	û 251	ü 252	ý 253	þ 254	ß 255	"F0
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,00000	x-height	4,30450 pt
spazio interparola	3,33251 pt	larghezza del quadrato	9,99755 pt
allungamento interparola	1,66625 pt	spazio extra	1,11083 pt
accorciamento interparola	1,11083 pt	corpo nominale	10,00000 pt

Figura 1.2: Il font latino a 256 caratteri con codifica T1

```
\else{\ooalign{\unhbox\z@\crcr
  \hidewidth\char11\hidewidth}}\fi}
```

ma si dice sempre che il comando per la cediglia `\c` trova il segno ‘cediglia’ nella posizione 11 per metterlo sotto la lettera che costituisce il suo unico argomento. Però per i segni con la cediglia presenti nella polizza codificata con la codifica *T1*, come ç, ţ, ş,[4] ulteriori comandi dicono che cosa fare per trovare il segno giusto nella polizza dei font.

Successivamente vengono definite le corrispondenze fra i simboli speciali e/o accentati e gli indirizzi in cui trovare questi segni nella polizza codificata *T1*. Per esempio:

```
\DeclareTextSymbol{\ss}{T1}{255}
...
\DeclareTextComposite{\"}{T1}{i}{239}
\DeclareTextComposite{\"}{T1}{\i}{239}
```

[4]Questi ultimi due, se usati in rumeno, non dovrebbero avere la cediglia, ma una virgola; spesso però i rumeni devono accontentarsi di una approssimazione, perché molti font mancano del segno con la virgola; esiste il pacchetto combelow per risolvere con LaTeX questo problema.

	’00	’01	’02	’03	’04	’05	’06	’07	’10	’11	’12	’13	’14	’15	’16	’17	
’000	Γ 0	Δ 1	Θ 2	Λ 3	Ξ 4	Π 5	Σ 6	Υ 7	Φ 8	Ψ 9	Ω 10	ff 11	fi 12	fl 13	ffi 14	ffl 15	"00
’020	ı 16	ȷ 17	` 18	´ 19	˘ 20	ˇ 21	¯ 22	° 23	¸ 24	ß 25	æ 26	œ 27	ø 28	Æ 29	Œ 30	Ø 31	"10
’040	˛ 32	! 33	” 34	# 35	$ 36	% 37	& 38	’ 39	(40	) 41	* 42	+ 43	, 44	- 45	. 46	/ 47	"20
’060	0 48	1 49	2 50	3 51	4 52	5 53	6 54	7 55	8 56	9 57	: 58	; 59	¡ 60	= 61	¿ 62	? 63	"30
’100	@ 64	A 65	B 66	C 67	D 68	E 69	F 70	G 71	H 72	I 73	J 74	K 75	L 76	M 77	N 78	O 79	"40
’120	P 80	Q 81	R 82	S 83	T 84	U 85	V 86	W 87	X 88	Y 89	Z 90	[91	“ 92	] 93	ˆ 94	˙ 95	"50
’140	‘ 96	a 97	b 98	c 99	d 100	e 101	f 102	g 103	h 104	i 105	j 106	k 107	l 108	m 109	n 110	o 111	"60
’160	p 112	q 113	r 114	s 115	t 116	u 117	v 118	w 119	x 120	y 121	z 122	– 123	— 124	” 125	˜ 126	¨ 127	"70
	"00	"01	"02	"03	"04	"05	"06	"07	"08	"09	"0A	"0B	"0C	"0D	"0E	"0F	

Parametri

inclinazione	0,00000	x-height	4,30554 pt
spazio interparola	3,33332 pt	larghezza del quadrato	10,19999 pt
allungamento interparola	1,66665 pt	spazio extra	1,11111 pt
accorciamento interparola	1,11111 pt	corpo nominale	10,00000 pt

Figura 1.3: Il font latino a 128 caratteri con codifica OT1

La prima dichiarazione serve per indicare che il segno ß ottenibile con il comando interno `\ss` si trova nella posizione 255 della polizza. Invece il segno ottenuto usando il comando per la dieresi con l’argomento `i` (i con il puntino) oppure `\i` (ı senza puntino) producono entrambi la ‘ï’ già accentata[5] che si trova nella posizione 239 della polizza dei font codificati *T1*.

Se invece non si è specificata nessuna opzione per la codifica del font di uscita, LaTeX usa la codifica di default *OT1*, tabella 1.3; per questa codifica aprendo il file *ot1enc.def* si trovano le seguenti dichiarazioni:

```
\DeclareTextAccent{\"}{OT1}{127}
\DeclareTextSymbol{\i}{OT1}{16}
\DeclareTextCompositeCommand{\"}{OT1}{i}{\"\i}
```

La prima dichiarazione indica dove trovare nella polizza dei caratteri il segno per la dieresi; la seconda dice dove trovare nella polizza dei caratteri la ‘i’ senza puntino; la terza dice che se in entrata si è scritto `\"i` invece che direttamente ï, la sequenza per il comando della dieresi seguita dalla lettera i col puntino deve essere interpretata alla stessa maniera di come si interpreta il codice di ingresso ï, cioè come `\"\i`. Come si vede, entrambe le dichiarazioni sono valide solo per la codifica *OT1* dei font di uscita.

Quando quindi LaTeX incontra una ‘i’ accentata o con la macro per l’accento o rappresentata direttamente con il codice d’entrata per la ‘i’ accentata, si ritrova a dover eseguire l’espressione `\"\i`; questa si traduce nei comandi primitivi che formano la stringa `\accent127\char16`; il comando primitivo `\accent` serve per sovrapporre il segno che

[5] Questo fatto succede con quasi tutti gli accenti quando l’argomento è una ‘i’. Per cui diventa obsoleta la prescrizione che per mettere un accento sopra la ‘i’ bisogna usare *solo* la variante senza puntino.

ha l'indirizzo specificato come suo argomento sul segno seguente che qui è indicato con il comando primitivo `\char` seguito dall'indirizzo specifico della 'i' senza puntino. Quei comandi primitivi verranno eseguiti solo quando il capoverso sarà stato elaborato sotto forma di righe giustificate con le eventuali cesure in fin di riga, pronto per essere conservato nella pagina in costruzione; ma quando l'algoritmo di divisione in righe entra in gioco, agisce sul testo che contiene ancora questi comandi primitivi. Come tali non fanno parte della stringa di token leciti in una parola, per cui l'algoritmo di sillabazione termina la 'parola' subito prima del comando per l'accento, cosicché non riesce a sillabare completamente l'intera stringa che gli umani considerano una parola.

Ora dovrebbe quindi essere chiaro che:

1. Quando pdflatex deve eseguire la sillabazione usando per l'uscita la codifica *T1* si trova nel suo buffer interno il capoverso da dividere in righe composto solo da caratteri, e non contiene comandi di nessun genere (tranne quelli che non riguardano il testo); e quindi funziona a dovere. Invece con la codifica *OT1*, quella di default, ogni comando per inserire gli accenti resta nel buffer del capoverso e abbiamo visto che questo impedisce a pdflatex di riconoscere come parole le stesse stringhe di caratteri che gli umani considerano come tali.

2. I segni accentati delle polizze codificate *T1* sono disegnati individualmente quindi lo stesso accento è collocato sulla lettera di base nella posizione più opportuna in relazione alla sua forma. Invece con la codifica *OT1* lo stesso accento viene sempre centrato sulla mezzeria della lettera di base senza tenere conto delle asimmetrie ottiche associate a ciascuna lettera di base.

Queste sono due ottime ragioni per non usare mai i font con codifiche diverse da *T1* quando si compongono con pdflatex testi in lingue che si scrivono in caratteri latini e che contengono diversi diacritici; in realtà questa raccomandazione vale anche per l'inglese, che solitamente non usa diacritici, ma li usa (*a*) nelle citazioni di nomi o parole straniere, e (*b*) nelle edizioni storiche o critiche di testi scritti in *middle English*.

Discorsi simili si possono fare per la codifica *utf8*, solo che ogni carattere UNICODE che si introduce con la tastiera attraverso lo *shell editor* nel file *.tex* non è formato da un solo byte, ma da un codice formato spesso da diversi byte, per cui la gestione dell'immissione dei caratteri nel file *.tex* è un poco più complicata; per fortuna ci pensa l'editor e l'utente è sollevato dalla necessità di occuparsi di questi dettagli. Anche la trasformazione dei caratteri immessi con eventuali comandi TEX o LATEX diventa un poco più complessa, ma ci pensa il programma di composizione. L'uso della codifica *utf8* offre però il vantaggio che, tastiera permettendo, qualunque segno immissibile con la tastiera può essere fatto corrispondere con il segno giusto, compatibilmente con la codifica dei font di uscita, cioè con l'effettiva presenza di quel segno nella polizza dei font di uscita. Se anche questi font sono codificati UNICODE, non c'è bisogno di nessun passaggio intermedio, ed è questo quello che fanno i compilatori xetex, luatex e context-mk-iv.

Capitolo 2

Specificare la codifica giusta

Qualsiasi sia la codifica dei caratteri di ingresso, nasce un problema che ogni utente di LaTeX, o degli altri programmi del sistema TeX, deve affrontare.

Egli deve specificare (mediante l'uso dell'opzione giusta al pacchetto inputenc nel preambolo del suo documento) quale sia la codifica con il quale ha scritto o modificato il file sorgente, altrimenti il suo file non potrà essere correttamente compilato.

Un errore comune è quello di scordarsi addirittura di invocare il pacchetto inputenc, non parliamo di specificare una opzione valida; oppure di invocare il pacchetto ma di specificare una opzione non coerente con le impostazioni dello *shell editor*. Succede più spesso di quanto non si creda. Tizio scrive un file sorgente *.tex* usando la codifica di ingresso *utf8*; poi spedisce il file al suo collega Caio. Caio, il cui *shell editor* è configurato per usare la codifica *latin1*, apre il file inviatogli da Tizio, ma trova il testo intercalato con caratteri strani. Questa è un situazione frequentissima.

È successo anche a chi scrive (utente di calcolatori Macintosh) di ricevere da un'altra persona, anch'essa utente di un Macintosh, un file LaTeX codificato con la codifica MacRoman (*applemac*); l'altra persona non si rendeva conto che quella è una codifica ristretta ai sistemi Mac e non è diffusa come le codifiche 'apolidi', non legate ad un particolare sistema operativo o a una marca di hardware. Chi scrive abitualmente usava la codifica ISO Latin 1 per scrivere in italiano e la codifica UTF-8 per scrivere con altri alfabeti; quando ha ricevuto il file ci ha messo un poco a rendersi conto di questa situazione e quando se ne è accorto era troppo tardi per porvi rimedio in modo corretto. Quanto segue, oltre a descrivere la situazione relativa ai malintesi generati dalle varie codifiche, serve anche per indicare alcuni modi corretti per affrontare questo tipo di problemi.

Se Tizio e Caio stanno usando lo *shell editor* multipiattaforma TeXworks e se hanno configurato bene il file *.tex* non devono fare nulla. Per configurare bene il file bisogna che esso, prima del testo vero e proprio, contenga delle righe di configurazione costituite da righe di commento, scritte in modo speciale, che dicono a TeXworks se esiste e come si chiama il master file e quale sia la codifica usata per scrivere il testo (nell'esempio che segue la prima riga è facoltativa, nel senso che se il file è il master file, non c'è nessun bisogno di fornire questa informazione lapalissiana)[1] e quale sia il programma che deve

[1] Le righe indicate qui di seguito seguono la sintassi di TeXShop; TeXworks ha un'altra sintassi, ma

essere usato per comporre il particolare file:

```
%␣!TEX␣root␣=␣MyMasterFile.tex
%␣!TEX␣encoding␣=␣UTF-8␣Unicode
%␣!TEX␣TS-program␣=␣pdflatex
```

In questo modo, se Tizio e Caio usano *shell editor* diversi, ma che siano in grado di decifrare queste speciali righe di commento, sono gli editor che si auto"configurano di conseguenza e i due tastieristi non devono fare assolutamente nulla.

Esistono poi alcuni *shell editor* che riconoscono automaticamente la codifica usata per creare i file. Apparentemente Aquamacs è uno di questi *shell editor*; ho provato a far leggere ad Aquamacs un file salvato con la codifica *applemac* (MacRoman) e sullo schermo è apparso il file con le lettere accentate sostituite dai loro codici ottali preceduti dal backslash; quindi a stretto rigore Aquamacs riconosce automaticamente molte codifiche, ma non riconosce "Western MacRoman" sebbene Aquamacs sia un programma specifico per le macchine Macintosh[2]; tuttavia la sostituzione globale di quei codici numerici ottali con le corrispondenti lettere accentate è molto facile.

L'editor Aquamcas dispone di altre righe speciali che seguono un'altra sintassi:

```
% -*- mode: latex; coding: latin-1;  TeX-master: MyMasterFile.tex -*-
```

Questa riga speciale deve essere all'inizio del file *.tex*; i due insiemi di righe speciali, quelle per Aquamacs/emacs e quelle per TEXShop"/TeXworks, possono coesistere nello stesso file in questo modo:

```
% -*- mode: latex; coding: latin-1;  TeX-master: MyMasterFile.tex -*-
% !TeX root = MyMasterFile.tex
% !TeX encoding = IsoLatin
% !Tex TS-program = pdflatex
```

e gli editor che le possono interpretare lo fanno senza confondersi con le righe presenti per editor diversi.

L'editor TeXstudio è in grado di distinguere la codifica *utf8*, e classifica come ISO Latin 1 ogni altra codifica; il che può essere errato e bisogna intervenire a mano.

Per impostare le righe speciali con TeXworks bisogna procedere a mano. Con TEXShop è più semplice perché il menù `Macros` contiene esplicitamente le voci `Encoding`, `Program` e `Root`; scegliendo ciascuna di queste voci si aprono pannelli di dialogo fra cui scegliere l'informazione giusta. Per Aquamacs alcune "variabili" della riga speciale devono essere introdotte a mano, ma per il nome del master file si può scegliere dal menù `LaTeX"/Multifile/Parsing"/Set Master File` che apre un pannello di dialogo dove scegliere il nome del file che deve svolgere il ruolo di master file.

---

capisce anche la sintassi di TEXShop.

[2] In effetti dando ad Aquamacs il comando per esporre le codifiche che può riconoscere, la codifica Western MacRoman non è elencata.

2.1 Scoprire la codifica di input

Se Tizio e Caio non si trovano in queste condizioni descritte nel paragrafo precedente, che cosa deve fare Caio per poter elaborare il testo contenuto nel file ricevuto da Tizio? Ha due possibilità: (*a*) cambia codifica al file *.tex*, oppure (*b*) cambia l'impostazione della codifica del suo *shell editor*. Ma prima di tutto deve scoprire quale sia la codifica del file ricevuto da Tizio.

La soluzione (*a*) sarebbe preferibile; e sarebbe preferibile anche che entrambi usassero sempre la codifica *utf8*. Se Tizio poteva compilare correttamente il suo file, era perché aveva specificato l'opzione giusta al pacchetto `inputenc`; Caio, allora, benché non riesca a leggere il testo del documento sorgente con facilità, può però leggere perfettamente il preambolo; basta allora che cerchi la stringa `inputenc` e lì trova che Tizio ha usato la codifica *latin1*. Chiude il file appena aperto, *ma senza salvarlo*, telefona a Tizio ricordandogli che avevano convenuto di usare entrambi la codifica *utf8*, riapre il file con TeXworks e procede come spiegato nel capitolo 3 salvando il file con la codifica *utf8*.

Fin qui va bene. Ma se Tizio ha usato la codifica *latin1* e si è scordato di specificare la codifica del suo *shell editor* nel preambolo del documento? Be', vuol dire due cose: primo, non poteva compilare il suo file producendo il file di uscita in modo corretto e, secondo, Caio deve tirare a indovinare per scoprire la codifica usata da Tizio, probabilmente in modo inconsapevole; è chiaro che basterebbe una telefonata o un messaggio di posta elettronica; è altrettanto chiaro che il procedimento indicato nel capitolo 3 permette di scoprire sia la codifica usata sia di eseguire la conversione. Ma talvolta il file è stato scritto da un anonimo Tizio che l'ha messo in rete e Caio l'ha trovato interessante guardandolo sullo schermo; ma dopo averlo scaricato non riesce più a leggerlo. In questo caso disperato, sempre senza modificare e salvare il file appena scaricato, tira ad indovinare; l'indovinello è facile se il file era stato scritto usando la codifica *utf8*, perché ogni carattere non ASCII è sostituito da due o più caratteri strani. Invece l'indovinello può essere difficile se appaiono caratteri strani isolati, specialmente alla fine delle parole. Non resta che seguire la procedura indicata nel capitolo 3 e non ci sono più problemi.

Il caso più disperato si ha quando Tizio e/o Caio si servono di *shell editor* vecchiotti, che usano una sola codifica, ma purtroppo si tratta di codifiche diverse. Non c'è quindi modo di modificare le impostazioni dello *shell editor* e non rimane altra scelta che quella di cambiare, mediante appositi programmi da linea di comando, la codifica del file *.tex*. O meglio, se i due collaboratori sanno di essere in queste condizioni avranno cura di scrivere o modificare i file *.tex* sorgente del loro lavoro cooperativo usando solamente caratteri ASCII; in sostanza non useranno direttamente i caratteri nazionali della loro tastiera, ma scriveranno le varie sequenze per gli accenti come richiesto dalla sintassi originale di TEX: per l'italiano useranno perciò `\‘a`, `\‘e`, `\’e`, `\‘i`, `\‘o`, `\‘u`, e si comporteranno in modo simile con le lettere maiuscole. È chiaro che questo modo di procedere, formalmente corretto, non è molto efficiente e i due collaboratori potrebbero accordarsi per aggiornare il loro sistemi TEX e per usare lo stesso fra uno dei vari *shell editor* multipiattaforma. In ogni caso seguendo la procedura indicata nel capitolo 3 possono entrambi convertirsi reciprocamente i file nel modo che vogliono. In fondo TeXworks è multipiattaforma e non è legato a nessuna particolare distribuzione del sistema TEX. Quindi se TeXworks

non fosse già installato, basta che entrambi i collaboratori lo installino. Possono anche usarlo per lavorare sui loro file, non solo per scoprire la codifica oppure per convertirne la codifica; è un editor semplice, ma ha diverse caratteristiche molto interessanti.

Si tenga presente che il modo apparentemente primitivo di scrivere i caratteri accentati secondo la sintassi TEX, consente molte più combinazioni di quelle che consentirebbe la codifica UNICODE, perché ogni segno diacritico di cui TEX dispone (una dozzina) può essere messo su qualunque lettera latina (26) minuscola o maiuscola, con un totale di combinazioni enorme, che consentono anche delle combinazioni "impossibili", cioè segni che non vengono usati in nessuna lingua; questo meccanismo non è affatto tanto primitivo come si potrebbe pensare, tant'è che costituisce la codifica interna del sistema TEX, indipendentemente dalla codifica di ingresso e da quella relativa ai font di uscita.

2.2 Cambiamento della codifica

Ma se proprio si deve cambiare codifica e, per qualche motivo, non si volesse, o non si potesse, o non si riuscisse a installare TeXworks tra i programmi disponibili, il metodo di base è è l'applicativo *iconv* per macchine Linux e Mac OS X (per macchine Windows si veda più avanti nella pagina 24), la cui sintassi è:

```
iconv [-c] [-s] [-f codifica] [-t codifica] inputfile
```

dove le opzioni `-c` e `-s` controllano i messaggi d'errore nel caso che il file di ingresso contenga caratteri non convertibili. Il file di ingresso `inputfile` è scritto con la codifica specificata con l'opzione `-f`, mentre il file di uscita è scritto con la codifica specificata con l'opzione `-t`. Il programma di default scrive il file di uscita nello *stream* di scrittura standard (generalmente lo schermo); se si vuole scrivere su un file vero e proprio bisogna reindirizzare l'uscita verso un file tramite il codice di reindirizzamento:

```
iconv [-c] [-s] [-f codifica] [-t codifica] inputfile > outputfile
```

dove il nome del file di uscita, `outputfile`, deve essere diverso dal nome del file di entrata.

Per conoscere le codifiche disponibili sulla particolare macchina sulla quale si sta lavorando e per scrivere i nomi delle codifiche con l'ortografia giusta, conviene dare il comando:

```
iconv -l
```

e sul terminale appare tutto l'elenco delle codifiche disponibili per le quali è possibile eseguire le conversioni. Le codifiche scritte sulla stessa riga, stando alla pagina del manuale, sono sinonimi l'una dell'altra. Nella macchina sulla quale sto lavorando le codifiche disponibili (con le righe ripiegate e rientrate per i limiti della giustezza del testo) sono:

```
ANSI_X3.4-1968 ANSI_X3.4-1986 \textsc{ascii} CP367 IBM367 ISO-IR-6
   ISO646-US ISO_646.IRV:1991 US US-\textsc{ascii} CS\textsc{ascii}
UTF-8
```

ISO-10646-UCS-2 UCS-2 CSUNICODE
UCS-2BE UNICODE-1-1 UNICODEBIG CSUNICODE11
UCS-2LE UNICODELITTLE
ISO-10646-UCS-4 UCS-4 CSUCS4
UCS-4BE
UCS-4LE
UTF-16
UTF-16BE
UTF-16LE
UTF-32
UTF-32BE
UTF-32LE
UNICODE-1-1-UTF-7 UTF-7 CSUNICODE11UTF7
UCS-2-INTERNAL
UCS-2-SWAPPED
UCS-4-INTERNAL
UCS-4-SWAPPED
C99
JAVA
CP819 IBM819 ISO-8859-1 ISO-IR-100 ISO8859-1 ISO_8859-1 ISO_8859-1:1987 L1 LATIN1 CSISOLATIN1
ISO-8859-2 ISO-IR-101 ISO8859-2 ISO_8859-2 ISO_8859-2:1987 L2 LATIN2 CSISOLATIN2
ISO-8859-3 ISO-IR-109 ISO8859-3 ISO_8859-3 ISO_8859-3:1988 L3 LATIN3 CSISOLATIN3
ISO-8859-4 ISO-IR-110 ISO8859-4 ISO_8859-4 ISO_8859-4:1988 L4 LATIN4 CSISOLATIN4
CYRILLIC ISO-8859-5 ISO-IR-144 ISO8859-5 ISO_8859-5 ISO_8859-5:1988 CSISOLATINCYRILLIC
ARABIC ASMO-708 ECMA-114 ISO-8859-6 ISO-IR-127 ISO8859-6 ISO_8859-6 ISO_8859-6:1987 CSISOLATINARABIC
ECMA-118 ELOT_928 GREEK GREEK8 ISO-8859-7 ISO-IR-126 ISO8859-7 ISO_8859-7 ISO_8859-7:1987 CSISOLATINGREEK
HEBREW ISO-8859-8 ISO-IR-138 ISO8859-8 ISO_8859-8 ISO_8859-8:1988 CSISOLATINHEBREW
ISO-8859-9 ISO-IR-148 ISO8859-9 ISO_8859-9 ISO_8859-9:1989 L5 LATIN5 CSISOLATIN5
ISO-8859-10 ISO-IR-157 ISO8859-10 ISO_8859-10 ISO_8859-10:1992 L6 LATIN6 CSISOLATIN6
ISO-8859-13 ISO-IR-179 ISO8859-13 ISO_8859-13 L7 LATIN7
ISO-8859-14 ISO-CELTIC ISO-IR-199 ISO8859-14 ISO_8859-14 ISO_8859-14:1998 L8 LATIN8
ISO-8859-15 ISO-IR-203 ISO8859-15 ISO_8859-15 ISO_8859-15:1998
ISO-8859-16 ISO-IR-226 ISO8859-16 ISO_8859-16 ISO_8859-16:2000

KOI8-R CSKOI8R
KOI8-U
KOI8-RU
CP1250 MS-EE WINDOWS-1250
CP1251 MS-CYRL WINDOWS-1251
CP1252 MS-ANSI WINDOWS-1252
CP1253 MS-GREEK WINDOWS-1253
CP1254 MS-TURK WINDOWS-1254
CP1255 MS-HEBR WINDOWS-1255
CP1256 MS-ARAB WINDOWS-1256
CP1257 WINBALTRIM WINDOWS-1257
CP1258 WINDOWS-1258
850 CP850 IBM850 CSPC850MULTILINGUAL
862 CP862 IBM862 CSPC862LATINHEBREW
866 CP866 IBM866 CSIBM866
MAC MACINTOSH MACROMAN CSMACINTOSH
MACCENTRALEUROPE
MACICELAND
MACCROATIAN
MACROMANIA
MACCYRILLIC
MACUKRAINE
MACGREEK
MACTURKISH
MACHEBREW
MACARABIC
MACTHAI
HP-ROMAN8 R8 ROMAN8 CSHPROMAN8
NEXTSTEP
ARMSCII-8
GEORGIAN-ACADEMY
GEORGIAN-PS
KOI8-T
MULELAO-1
CP1133 IBM-CP1133
ISO-IR-166 TIS-620 TIS620 TIS620-0 TIS620.2529-1 TIS620.2533-0
TIS620.2533-1
CP874 WINDOWS-874
VISCII VISCII1.1-1 CSVISCII
TCVN TCVN-5712 TCVN5712-1 TCVN5712-1:1993
ISO-IR-14 ISO646-JP JIS_C6220-1969-RO JP CSISO14JISC6220RO
JISX0201-1976 JIS_X0201 X0201 CSHALFWIDTHKATAKANA
ISO-IR-87 JIS0208 JIS_C6226-1983 JIS_X0208 JIS_X0208-1983
JIS_X0208-1990 X0208 CSISO87JISX0208

ISO-IR-159 JIS_X0212 JIS_X0212-1990 JIS_X0212.1990-0 X0212
CSISO159JISX02121990
CN GB_1988-80 ISO-IR-57 ISO646-CN CSISO57GB1988
CHINESE GB_2312-80 ISO-IR-58 CSISO58GB231280
CN-GB-ISOIR165 ISO-IR-165
ISO-IR-149 KOREAN KSC_5601 KS_C_5601-1987 KS_C_5601-1989
CSKSC56011987
EUC-JP EUCJP EXTENDED_UNIX_CODE_PACKED_FORMAT_FOR_JAPANESE
CSEUCPKDFMTJAPANESE
MS_KANJI SHIFT-JIS SHIFT_JIS SJIS CSSHIFTJIS
CP932
ISO-2022-JP CSISO2022JP
ISO-2022-JP-1
ISO-2022-JP-2 CSISO2022JP2
CN-GB EUC-CN EUCCN GB2312 CSGB2312
CP936 GBK
GB18030
ISO-2022-CN CSISO2022CN
ISO-2022-CN-EXT
HZ HZ-GB-2312
EUC-TW EUCTW CSEUCTW
BIG-5 BIG-FIVE BIG5 BIGFIVE CN-BIG5 CSBIG5
CP950
BIG5-HKSCS BIG5HKSCS
EUC-KR EUCKR CSEUCKR
CP949 UHC
CP1361 JOHAB
ISO-2022-KR CSISO2022KR

Perciò per trasformare il file *myfile.tex* scritto con la codifica *latin1* in un altro file *myfile-utf8.tex* scritto con la codifica *utf8* bisogna dare il comando:

```
iconv -f LATIN1 -t UTF-8 myfile.tex > myfile-utf8.tex
```

Per convertire un file *myfilew.tex* scritto su una macchina Windows con la *code page* di default 1252 in un file con la codifica *latin1* si scriverà:

```
iconv -f WINDOWS-1252 -t LATIN1 myfilew.tex > myfile-latin1.tex
```

Per il Mac esiste un interfaccia grafica *charco* che permette di fare sostanzialmente le stesse cose lavorando prevalentemente con il mouse.

Per le macchine Windows si può usare ancora *iconv* ma bisogna installarlo dentro l'ambiente di simulazione CygWin operando come se si stesse installando il programma su una macchina Linux.

Alternativamente si può scaricare la libreria libiconv dal sito http:/"/sourceforge.net"/projects"/gnuwin32" e il programma di installazione provvede anche a rendere disponibile la versione Windows *Win-iconv* con la quale si può operare senza ricorrere all'ambiente di simulazione CygWin.

Se tutto ciò dovesse 'disturbare' o comunque apparire troppo complicato, si riesamini la possibilità di sfruttare le capacità dello *shell editor* TeXworks, oppure si ricorra alla sintassi TEX che non fa uso di caratteri speciali ma solo caratteri ASCII e usa macro particolari per indicare quali segni diacritici usare.

2.3 Collage di contributi diversi

Un altro problema sorge invece quando si compone un testo eseguendo un collage di contributi di varie persone che hanno fornito file in formati come *.doc* o *.pdf*, e il compositore procede a mano copiando dal file esterno e incollando nel file *.tex*.

In questi casi spesso sono i caratteri di interpunzione quelli che differiscono sensibilmente dai caratteri della codifica che si vorrebbe usare, fosse essa anche la codifica più generale *utf8*. A occhio nudo spesso questi caratteri (apostrofo, virgolette semplici, virgolette diritte, eccetera) sono difficili da rilevare perché il riconoscimento corretto dipende anche dal disegno dei font che vengono usati dallo *shell editor* nella finestra del file sorgente; io preferisco i font a spaziatura fissa in questa finestra del file sorgente ma, nonostante questo, quei segni sono difficili da rilevare ad occhio nudo. Tenendo conto che un file di ingresso deve essere scritto tutto con la stessa codifica, bisogna per forza convertire a mano tutti i segni nella codifica principale, oppure marcare il file sorgente con la codifica specifica usata dal collaboratore esterno o che il suo programma di trasformazione nel file *.pdf* ha usato.

TeXworks e TEXShop potrebbero interpretare correttamente questa informazione e Aquamacs potrebbe diagnosticare da solo la codifica usata. In ogni caso si può eseguire la compilazione con pdflatex e poi andare a controllare un certo numero di istanze di questi caratteri speciali passando dalla finestra del file sorgente alla finestra del file composto sfruttando la capacità dello *shell editor* e del visualizzatore del file composto di eseguire correttamente la ricerca diretta e inversa. Se questi caratteri un po' particolari sono tutti presenti in modo corretto nel file di uscita, allora la codifica dichiarata o determinata è corretta. Altrimenti è opportuno procedere ad una correzione manuale di tutti questi segni; solitamente la ricerca e sostituzione di *tutte* le istanze di un dato carattere può essere fatta con pochi click di mouse con qualunque *shell editor*.

Capitolo 3

Considerazioni riassuntive

Vale la pena di seguire la *check list* che segue, ma prima è opportuno rispondersi a questa domanda: "Che codifica di ingresso voglio usare come codifica predefinita?" Considerati i vantaggi (molti) e gli svantaggi (pochi: esistono ancora alcuni editor che non sono in grado di gestire la codifica *utf8*) io consiglierei di usare sempre la codifica *utf8*. Di solito questa è la codifica preimpostata quando si installa un editor moderno. Ma se si rispondesse che si è più interessati alla codifica *latin1*, per esempio, si abbia l'accortezza di impostarla per tutti gli editor che si desidera usare; se si usano file memorizzati su disco con altre codifiche, non si cambi l'impostazione dell'editor preferito, ma si proceda al cambiamento di codifica del file salvandolo con la codifica che si preferisce. Non bisogna scordarsi che se si riapre il file con lo stesso editor, il file (salvato correttamente cioè "senza errori") sullo schermo potrebbe nuovamente sembrare che contenga dei caratteri strani; questo è solo dovuto al fatto che in generale l'editor è preconfigurato per una codifica diversa: per esempio, l'editor è configurato per usare la codifica *utf8*, il file di cui convertire la codifica aveva la codifica *applemac* e lo si è convertito con la codifica *latin1* senza problemi e lo si è salvato; lo si riapre con lo stesso editor e, per bacco!, ci sono di nuovo i caratteri strani; è ovvio, perché il file è in *latin1* ma l'editor è configurato per *utf8*. L'ho detto fin dall'inizio che questo è il problema che si affronta in questa guida, quindi ora dovrebbe essere chiaro il perché di questo "fenomeno". È per questo che ho anche consigliato di scrivere in testa al file la o le righe di autoconfigurazione per l'editor: se questo è capace di interpretarle si autoconfigura. Ma è opportuno sottolineare che quelle righe servono *anche* per l'utente perché si ricordi del problema della consistenza fra la codifica del file e la codifica preimpostata per l'editor. In gin modo l'utente non si scordi di verificare e, eventualmente, di modificare l'opzione per il pacchetto inputenc, che deve sempre essere unica per tutti i file che costituiscono la collezione di file sorgente per un dato documento e deve essere conforme alla codifica con coi tutti i file sono stati salvati.

- La codifica di ingresso deve essere sempre specificata come opzione nell'invocazione del pacchetto inputenc; questa codifica riguarda essenzialmente l'interazione fra la tastiera e lo *shell editor* e come questo salva il file sorgente sul disco. Il programma di composizione, quello che usa questa informazione, non ha altri metodi per conoscere la codifica dei file sorgente che sta elaborando.

- Per la perfetta composizione del documento finale bisogna sempre usare font il più possibile completi, comunque almeno con la codifica *T1* per migliorare la divisione in sillabe in fin di riga e per avere i segni accentati di fattura migliore rispetto a quelli che si potrebbero avere sovrapponendo l'accento alla lettera di base.
- È fortemente consigliabile scrivere le righe di autoconfigurazione per lo *shell editor*; questo vale anche se non si usa uno *shell editor* che le capisca e possa autoconfigurarsi di conseguenza; esse rappresentano un buon promemoria permanente e sono utilissime per l'utente e per le altre persone a cui eventualmente si debba inviare il file sorgente.
- Se si deve cambiare codifica, allora conviene procedere come segue.
  1. È necessario eseguire subito una copia di backup del file di cui si deve cambiare la codifica; nel caso ci si sbagli con le operazioni successive, il file originale può sempre essere recuperato.
  2. Se aprendo un file *.tex* con lo *shell editor* si vede che il file contiene dei caratteri strani, allora l'impostazione della codifica dello *shell editor* non corrisponde alla codifica con cui è stato scritto il file *.tex*. In questo caso si eviti di fare qualunque modifica al file e si chiuda il file facendo attenzione a *non* "salvarlo" su disco[1].
  3. Si riapra il file con TeXworks; questo editor è disponibile per tutte le piattaforme e solo con le installazioni Linux potrebbe essere necessario installarlo espressamente. Per le operazioni che seguono è essenziale usare TeXworks, poi volendo si può tornare a usare qualunque altro editor che l'utente preferisca rispetto a TeXworks.
  4. Una volta aperto il file che contiene caratteri strani, ci si porti su una pagina in cui se ne vedono alcuni, sostituiti, generalmente, da rombi neri contenenti all'interno un punto interrogativo bianco; si mantenga quella pagina.
  5. Nella riga inferiore della schermata di TeXworks ci sono in basso a destra delle finestrine, nella seconda delle quali c'è la codifica impostata di default per questo editor; cliccando su questa finestra si apre il lungo elenco di codifiche che TeXworks è capace di gestire; si scelga un'altra codifica: per esempio, se inizialmente nella finestrina c'era scritto UTF8, si scelga ISO8859-1, poi si clicchi di nuovo sulla finestrina e nella prima riga in alto si scelga di "ricaricare lo stesso file con la codifica appena scelta" (fare attenzione a specificare nella finestra di dialogo di voler riaprire il file senza salvarlo).
  6. Il file viene ricaricato: se i rombi neri spariscono, quella appena scelta è la codifica con cui era stato scritto il file; se i rombi neri non spariscono ripetere scegliendo un altra codifica come indicato nel passo precedente. Si noti che le codifiche più frequenti sono ISO8859-1, MacRoman, Windows 1250, oltre a UTF-8 che è la codifica preimpostata di TeXworks.

[1] La maggior parte degli editor moderni hanno un una voce di menù, o uno shortcut di tastiera, o una piccola icona per chiudere un file e, se il file ha subito modifiche, l'editor chiede se si vogliono salvare le modifiche: rispondere di 'no' per chiudere senza salvare le modifiche.

7. Se si vuole cambiare codifica (come si è supposto all'inizio di questa elencazione), si scelga la codifica desiderata nella solita finestrina e questa volta si salvi il file con questa codifica.

- In questo modo si è scoperta la codifica del file iniziale e lo si è convertito nella codifica desiderata; se si dispone di TEXShop si apra il file appena salvato e con i suoi pulsanti gli si faccia scrivere in testa al file la riga magica dell'*encoding* corrispondente alla codifica appena usata per la conversione. In questo modo non ci si dovrà più preoccupare di capire"/indovinare la codifica ignota di quel file.

- Non ci si dimentichi assolutamente di controllare che la nuova codifica del file sia coerente con l'opzione passata al pacchetto inputenc; ne non lo fosse si modifichi l'opzione coerentemente. Certo se ci si dimenticasse di fare questa operazione essenziale, tutto il lavoro fatto sarebbe totalmente inutile.

- Se il documento da comporre è formato da un master file e da diversi file inclusi con `\input` o con `\include`, ci si ricordi che l'opzione di codifica per la composizione con uno dei programmi del sistema TEX viene specificata una volta sola, e il master file nonché tutti i file inclusi devono essere composti con la stessa codifica; va bene mettere le righe di autoconfigurazione in testa ad ogni file (servono per lo *shell editor*, se è capace di interpretarle, non servono per il programma di composizione, a cui invece serve l'opzione corretta specificata al pacchetto inputenc), ma siccome sono per comporre un unico documento esse devono tutte coincidere con la stessa opzione al pacchetto inputenc, opzione che non può venire modificata da un file all'altro ma, lo si ripete, è specificata una volta sola.

Che strategia seguire per ridurre al minimo i problemi derivanti dalla codifica? Io suggerisco caldamente di usare la codifica di entrata *utf8*; i file sorgente aumentano di dimensioni in maniera trascurabile, ma si è sicuri che, pur di usare uno *shell editor* recente che accetti la codifica di ingresso *utf8*, non esistono problemi di codifica a livello di pdflatex. Va anche notato che le versioni aggiornate dei più moderni *shell editor* già dalla prima installazione sono preimpostati per lavorare con la codifica *utf8*.

A chi si avvicina al sistema TEX per la prima volta, oppure aggiorna la propria distribuzione del sistema TEX dopo anni di mancati aggiornamenti, consiglio di installare, se non lo sono già, uno o più degli editor multipiattaforma TeXworks, TeXstudio, TeXmaker, di impostarli, se non lo fossero già, per lavorare con la codifica di ingresso *utf8*, e di mantenere sempre questa codifica. Se capitasse loro di dover aprire un file ricevuto da terzi o scaricato dalla rete che non fosse scritto con questa codifica, procedano come detto sopra per convertire il file alla codifica *utf8*.

Tuttavia la codifica *utf8* non garantisce di evitare problemi quando si esegue lavoro cooperativo, visto che i vari 'tastieristi' non necessariamente dispongono degli stessi *shell editor* operanti su macchine dello stesso genere, controllate dallo stesso sistema operativo. Bisogna sempre essere pronti a riconoscere i problemi della codifica di entrata e saperli risolvere. Questa raccomandazione è particolarmente indicata per coloro che si sono avvicinati al sistema TEX da poco; oppure a coloro che aggiornano il loro strumenti

software a versioni più moderne e non si aspettano di dover affrontare questo genere di novità.

Bibliografia

Beccari, C. (2012). «Introduzione all'arte della composizione tipografica con LaTeX». PDF document. URL http://www.guitex.org/home/.

Berry, K. (2011). «The TeX Live guide». PDF document. Leggibile con `texdoc texlive` nella distribuzione TeX Live.

Gregorio, E. (2010). «LaTeX, breve guida ai pacchetti di uso più comune». PDF document. URL http://profs.sci.univr.it/~gregorio/egtex.html.

Kopka, H. e Daly, P. W. (2004). *Guide to LaTeX*. Addison Wesley, 4ª edizione.

Lamport, L. (1994). *LaTeX: A document prparation system – User guide and reference manual*. Addison Wesley, 2ª edizione.

Mittelbach, F. *et al.* (2006). «LaTeX font encodings». PDF document. Leggibile con `texdoc encguide` nella distribuzione TeX Live.

Pantieri, L. e Gordini, T. (2012). «L'arte di scrivere con LaTeX». PDF document. URL http://www.guitex.org/home/.