Codifica delle informazioni: testi e colori

Angelo Di Iorio

Università di Bologna

Codifica di contenuti e dati Web

- Prima di visualizzare ed elaboratore contenuti e dati Web, si pone il problema di rappresentarli all'interno di un calcolatore e trasferirli da un calcolatore all'altro
- Il calcolatore infatti usa una rappresentazione numerica dell'informazione e in particolare la **codifica binaria, basata su due simboli 0 e 1 (bit)**
- Per rappresentare un dato non numerico abbiamo quindi la necessità di **trasformarlo in una sequenza di numeri** memorizzabili nel calcolatore
- Questo processo di **digitalizzazione** varia a seconda del dato che vogliamo rappresentare
- Noi guardiamo testi e colori ma ragionamenti simili si applicano a suoni e video

Background: codifica binaria ed esadecimale

- Il sistema di numerazione decimale che utilizziamo correntemente è un sistema cosiddetto ***posizionale***, ovvero dove conta la posizione di ogni cifra rispetto alle altre.
  - Ad esempio, in base dieci esistono dieci valori che possono essere rappresentati con una sola cifra, per i valori più grandi di 9 si accostano tra loro più cifre.
- A seconda della posizione delle cifre, queste assumono un significato diverso
- Lo stesso sistema si può applica usando basi diverse, e in particolare:
  - **Codifica binaria (0b)**: due simboli 0 e 1
  - **Codifica esadecimale (hex, 0x)**: 16 simboli 0, 1, 2, …, 9, A, B,C, D, E, F

Codifica binaria

- Il numero 317 in base dieci può essere interpretato come 3 *centinaia*, 1 *decina* e 7 *unità* ossia:

$$3*10^2 + 1*10^1 + 7*10^0 = 3*100 + 1*10 + 7*1$$

- Allo stesso modo si interpreta un numero in base due.

- Ad esempio:

$$1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 =$$

$$1*2^5 + 0*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 1*2^0 =$$

$$32 + 0 + 0 + 4 + 0 + 1 = 37$$

Codifica esadecimale

- Allo stesso modo si interpreta un numero in base sedici
- Oltre alle cifre decimali sono ammessi i caratteri A (che corrisponde a 10), …. , F(che corrisponde a 15).
- Ad esempio:

$$\mathtt{A\ \ 2\ \ 8\ =}$$

$$10*16^2 + 2*16^1 + 8*16^0 =$$

$$2560 + 32 + 8 = 2600$$

Codifica binaria ed esadecimale

- Poiché 16 = 2^4, c'è una connessione diretta tra rappresentazione binaria e esadecimale dei numeri secondo cui ogni blocco di 4 cifre binarie (bit) corrisponde ad una cifra esadecimale.
- Quindi un byte composto da 8 cifre binarie è composto da due cifre esadecimali:
  - Per esempio, 0b11001110 può essere diviso in 1100-1110
  - poiché 0b1100 = 0xC e 0b1110 = 0xE
  - 0b11001110 = 0xCE (= 12*16 + 14 = 206 in decimale)
- I valori in binario si iniziano con 0b e quelli in esadecimale con 0x

Quanti simboli?

- *k bit => 2^k stati => 2^k simboli/oggetti*
- Ricapitolando:
  - 1 bit => 2 valori (0, 1)
  - 2 bit => 2^2 valori (00, 01, 10, 11)
  - 7 bit => 2^7 valori => 128 valori
  - 8 bit (1 byte) => 2^8 valori => 256 valori
  - 16 bit (2 byte) => 2^{16} valori => 65536 valori
  - 32 bit (4 byte) => 2^{32} valori => 4.294.967.296 valori

Codifica testi e caratteri

- Per codificare un testo si **traduce ogni carattere in un valore numerico**, rappresentato poi in notazione binaria o esadecimale, e si **concatenano i valori numerici ottenuti**
- Deve essere evidente e non ambiguo il criterio di associazione di un blocco di bit ad un carattere dell'alfabeto e il meccanismo di traduzione
- Sono stati proposti diversi standard e set di caratteri per gestire le diversità dei vari alfabeti
- Noi ne vediamo solo alcuni (e non in dettaglio)

ASCII

(American Standard Code for Information Interchange)

- standard ANSI (X3.4 - 1968) che definisce valori per 128 caratteri, ovvero 7 bit su 8. Nello standard originale il primo bit non è significativo ed è pensato come bit di parità.
- ASCII possiede 33 caratteri (0-31 e 127) di controllo, inclusi alcuni ormai non più rilevanti e legati all'uso di telescriventi negli anni '60. Codifica anche spazi e "andata a capo" (CR e LF)
- Gli altri 95 sono caratteri dell'alfabeto inglese, maiuscole e minuscole, numeri e punteggiatura.

ASCII

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
010	sp	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
011	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
101	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
110	`	a	b	c	d	e	f	g	h	I	j	k	l	m	n	o
111	p	q	r	s	t	u	v	w	x	Y	z	{	|	}	~	canc

c è codificato in: 1100011 - 99 (decimale) - 0x63 (esadecimale)

C è codificato in: 1000011 - 67 (decimale) - 0x43 (esadecimale)

ISO 8859/1 (ISO Latin 1)

- Diverse estensioni di ASCII sono state proposte per utilizzare il primo bit e accedere a tutti i 256 caratteri. Nessuna di queste è standard tranne ISO Latin 1
- ISO 8859/1 (ISO Latin 1) è comprende un certo numero di caratteri degli alfabeti europei come accenti, ecc.
- ISO Latin 1 è usato automaticamente da HTTP e alcuni sistemi operativi.
- Ovviamente ISO Latin 1 è compatibile all'indietro con ASCII, di cui è un estensione per i soli caratteri >127.

ISO Latin 1 e windows-1252

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
80			‚	ƒ	„	…	†	‡	ˆ	‰	Š	‹	Œ			
90		‘	’	“	”	•	–	—	˜	™	š	›	œ			Ÿ
A0		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
B0	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

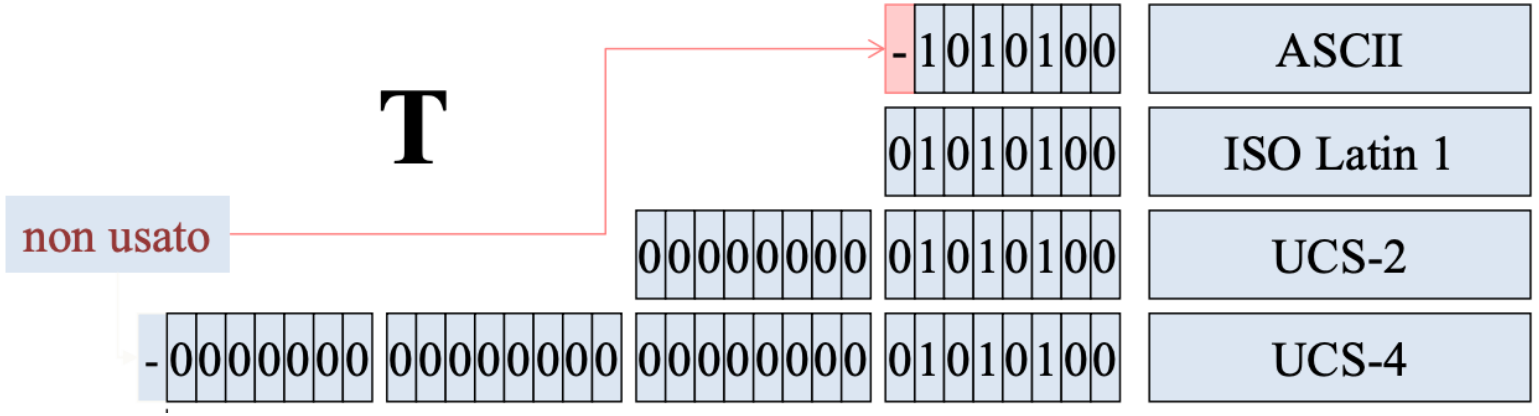
- Nota: righe e colonne sono indicate in notazione esadecimale. Il codice si ottiene dalla somma riga+colonna (es. C = 0x43)

UCS e UTF

- Parallelamente alla codifica dei caratteri latini, ci sono state analoghe codifiche per caratteri del cirillico, del greco, dell'ebraico
- Il discorso si complica per i CJKV (giapponesi, cinesi, coreani e vietnamiti) di origine cinese e per cui non bastano 8 bit.
- Si è cercato di definire a uno standard unico in grado di coprire tutti gli alfabeti che quindi usa un numero maggiore di byte. Anche qui hanno lavorato diversi gruppi che sono riusciti poi a convergere
- Due grandi famiglie di schemi:
  - a **lunghezza fissa**: UCS-2 (2 byte) e UCS-4 (4 byte)
  - a **lunghezza variabile**: UTF-8, UTF-16 e UTF-32

UCS e lunghezza fissa

- UCS-2 è uno schema a due byte ed è un'estensione di ISO Latin 1
- UCS-4 è uno schema a 31 bit in 4 byte, estensione di UCS-2
- Si usa sempre lo stesso numero di byte per ogni carattere (due o quattro se UCS-2 o UCS-4)

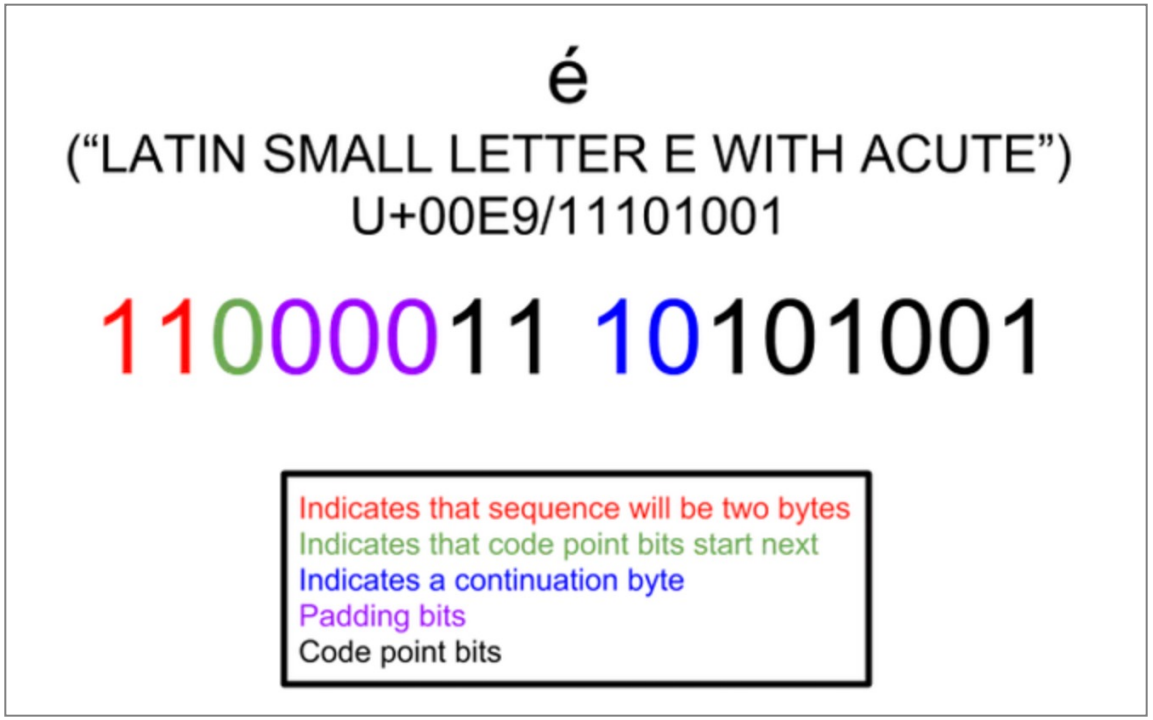


UTF-8

- UTF nasce dall'osservazione che i testi nella maggior parte dei casi sono scritti in un unico alfabeto o in alfabeti vicini
- E' quindi uno spreco usare 2 o 4 byte per ogni carattere, anche quando sarebbe sufficiente usare 1 byte
- UTF (Unicode Transformation Format o UCS Transformation Format) è un sistema a lunghezza variabile che permette di accedere a tutti i caratteri definiti di UCS-4:
  - I codici compresi tra 0 - 127 (ASCII a 7 bit) richiedono 1 byte
  - I codici derivati dall'alfabeto latino e tutti gli script non-ideografici richiedono 2 byte
  - I codici ideografici (orientali) richiedono 3 byte
  - Tutti gli altri 4 byte.
- Bit iniziali e di controllo permettono di capire come interpretare la restante seguenza di bit (quanti byte considerare)

Un esempio

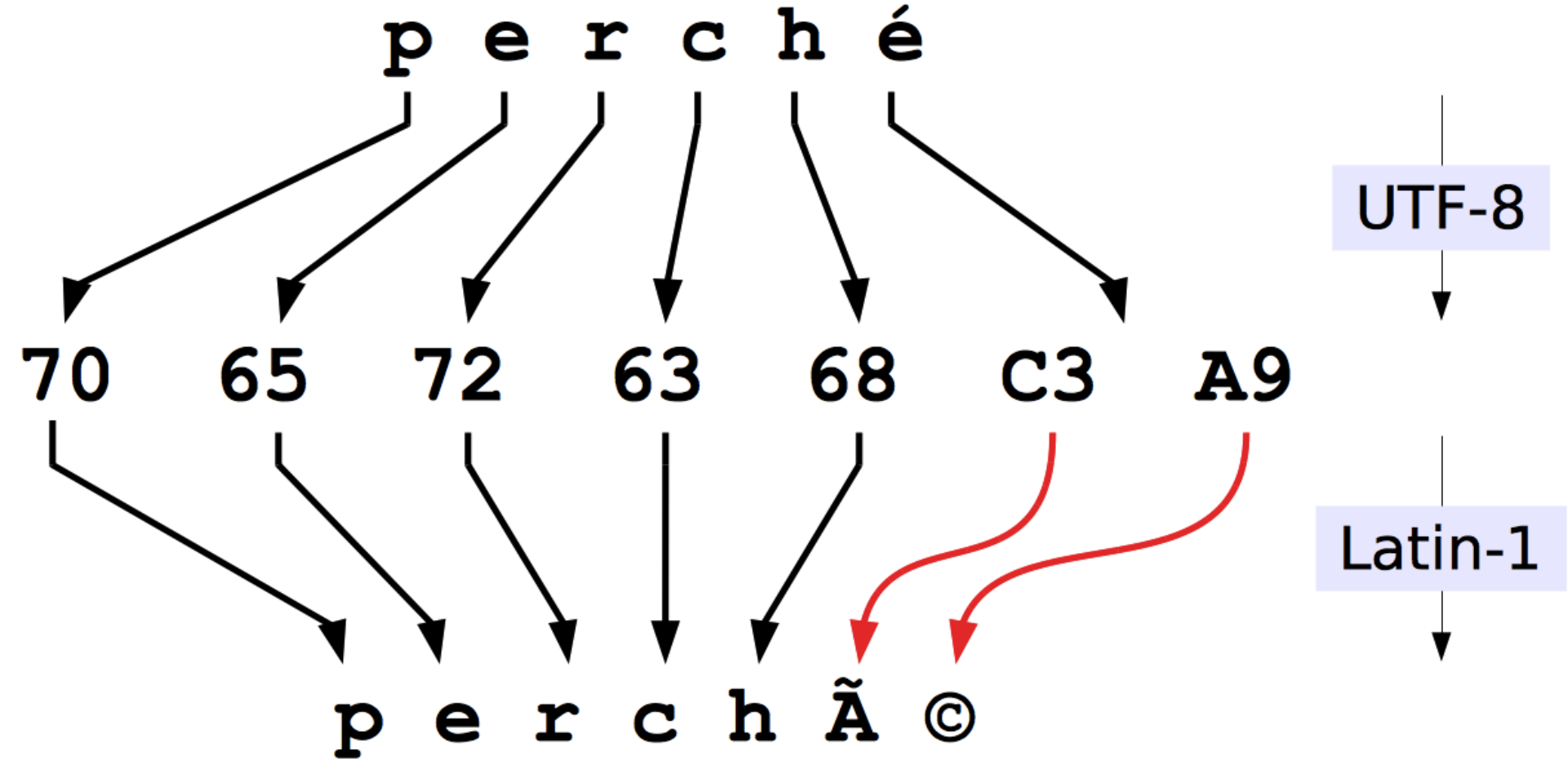
- La lettera è occupa due byte essendo un simbolo dell'alfabeto latino



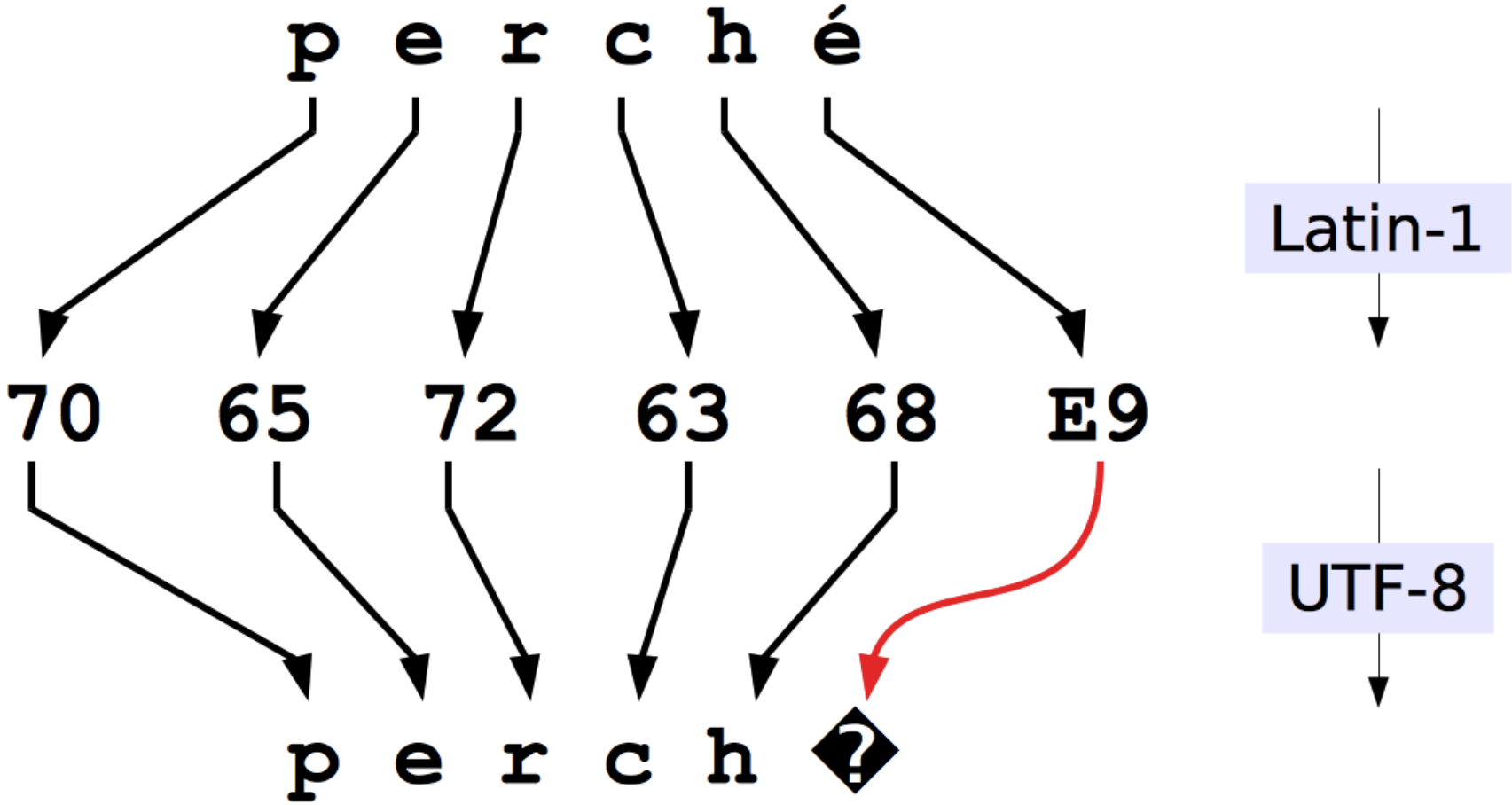
Differenze tra UTF-8 e ISO Latin-1

- Non confondere le codifiche UTF-8 e ISO Latin-1!
- Per i caratteri appartenenti ad ASCII, le due codifiche sono identiche. Quindi un documento in inglese non avrà differenze nel testo
- Viceversa, un testo in italiano, o francese o tedesco risulta quasi corretto, perché non vengono descritte correttamente solo le decorazioni di lettere latine (ad esempio, accenti, umlaut, vocali scandinave ecc.).
- In questo caso, UTF-8 utilizza 2 byte per questi caratteri, mentre ISO Latin 1 ne usa uno solo

Errori comuni



Errori comuni



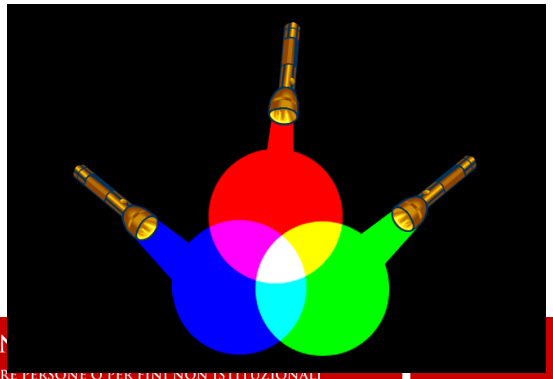
Codifica colori

Come codificare i colori?

- L'occhio umano percepisce i colori attraverso tre tipi di coni, che contengono pigmenti sensibili a diverse lunghezze d'onda. Queste frequenze sono interpretate separatamente e la combinazione di questi stimoli ci permette di percepire il colore
- Lo spettro complessivo dei colori riconoscibili dall'occhio umano può essere espresso come una spazio lineare di valori organizzati su un numero di dimensioni comode (3 o 4)
- In realtà questo spazio non è omogeneo e uniforme, la percezione dei colori è molto complessa
- I colori vengono riprodotti per "sintesi" con due approcci:
  - Spazi colore **additivi**
  - Spazi colore **sottrattivi**

Sintesi additiva

- In uno spazio colore additivo, ogni colore è definito come la ***somma*** del contributo di tre o quattro colori primari.
- Il nero è definito come l'assenza di contributi, poi i colori diventano via via più chiari e brillanti tanto maggiore è il contributo di ciascuna componente. Il bianco è definito come il massimo contributo possibile di tutte le componenti.
- I device ad emissione di luce, come gli schermi dei computer o i proiettori, definiscono i colori usando un modello additivo
- Gli stimoli arrivano nell'occhio simultaneamente e in rapida successione e fanno percepire il colore



RGB

- RGB è uno spazio colore additivo basato sull'identificazione di *Rosso*, *Verde* e *Blu* come colori primari. E' un modello ragionevolmente vicino a quello dell'occhio umano.
- Il colore è quindi espresso con 3 valori. In base al numero di bit usato per ogni valore si possono esprimere tonalità diverse di quel colore
- Con 1 byte per colore (RGB24), si avranno quindi 256 valori per ogni colore prima (esprimibili in notazione decimale, binaria ed esadecimale)

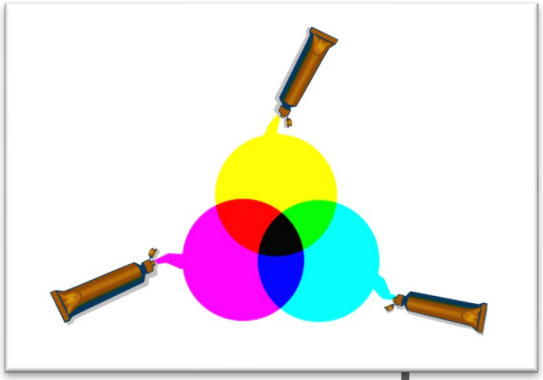
0, 0, 0	255, 0, 255	255, 127, 0
255, 255, 255	0, 255, 255	127, 64, 0
255, 0, 0	0, 255, 0	127, 127, 0
255, 255, 0	0, 0, 255	127, 127, 127

RGBa

- RGBa è uno spazio colore derivato dav RGB in cui viene aggiunta una quarta dimensione, chiamata *canale alpha*.
- Il canale alpha esprime come percentuale 0-100% l'opacità (ovvero la non trasparenza) con cui il colore RGB corrispondente lascia trapelare all'occhio umano la tinta sottostante.
- Nell'esempio l'immagine è coperta da rettangoli gialli con opacità varia, da `RGBa(255,255,0,0)` a `RGBa(255,255,0,100)`

Sintesi sottrattiva

- In uno spazio colore sottrattivo, ogni colore è definito come lo spettro residuo della luce ambientale riflessa da una composizione di pigmenti di colori primari che bloccano (***sottraggono***) parzialmente tale riflesso.
- Secondo questo modello, il bianco è definito come l'assenza di contributi (cioè è il colore della superficie riflettente senza pigmenti) e il nero è il colore raggiunto coprendo completamente la sorgente riflettente con pigmenti.
- La mescolanza di basa su meccanismi fisici ed è usata dalle stampanti
- Lo spazio colore sottrattiva più usato si chiama CMY (Cyan – Magenta – Yellow)



CMY(K)

- CYMK è uno spazio colore sottrattivo basato sull'identificazione di quattro colori primari, Cyan, Giallo, Magenta e Key (che è un nero molto scuro).
- CYMK usa quattro colori invece di tre perché l'inchiostro nero fornisce una definizione dei colori molto migliore, aumentando il contrasto e riducendo la quantità di inchiostro colorato necessario per le tinte più scure.
- Nei sistemi a stampa fotografica, i colori sono applicati sulla carta sulla base di quattro pellicole indipendenti ciascuna per uno dei quattro colori, e quella nera è quella in cui i dettagli e le forme dell'immagini sono meglio riconoscibili, quindi è la pellicola chiave per riconoscere l'immagine.
- Anche qui si esprime (in notazione binaria, decimale o esadecimale) la quantità di ogni colore da sottrarre

Conclusioni

- Per poter essere elaborati e trasmessi i contenuti e i dati su Web devono essere codificati come valori numerici, espressi in notazione binaria ed esadecimale
- Abbiamo visto come codificare testi, in alfabeti diversi, e colori
- Immagini, suoni e video richiedono processi di digitalizzazione specifici che qui non vediamo
- Attenzione alla codifica caratteri nei messaggi HTTP, nei documenti e nelle applicazioni