Web Applications
Text File Format

Martin Caminada

Cardiff University

Plain Text File Format

Example file:

```
Hello world!
How are you?
```

Plain Text File Format

Example file:

```
Hello world!
How are you?
```

```
H   e   l   l   o       w   o   r   l   d

!   \n  h   o   w       a   r   e       y

o   u   ?
```

Plain Text File Format

Example file:

```
Hello world!
How are you?
```

ASCII
character
encoding

```
H    e    l    l    o         w    o    r    l    d
72   101  108  108  111  32   119  111  114  108  100

!    \n   h    o    w         a    r    e         y
33   10   104  111  119  32   97   114  101  32   121

o    u    ?
111  117  63
```

Hex	Dec	Char		Hex	Dec	Char	Hex	Dec	Char	Hex	Dec	Char
0x00	0	NULL	null	0x20	32	Space	0x40	64	@	0x60	96	`
0x01	1	SOH	Start of heading	0x21	33	!	0x41	65	A	0x61	97	a
0x02	2	STX	Start of text	0x22	34	"	0x42	66	B	0x62	98	b
0x03	3	ETX	End of text	0x23	35	#	0x43	67	C	0x63	99	c
0x04	4	EOT	End of transmission	0x24	36	$	0x44	68	D	0x64	100	d
0x05	5	ENQ	Enquiry	0x25	37	%	0x45	69	E	0x65	101	e
0x06	6	ACK	Acknowledge	0x26	38	&	0x46	70	F	0x66	102	f
0x07	7	BELL	Bell	0x27	39	'	0x47	71	G	0x67	103	g
0x08	8	BS	Backspace	0x28	40	(	0x48	72	H	0x68	104	h
0x09	9	TAB	Horizontal tab	0x29	41	)	0x49	73	I	0x69	105	i
0x0A	10	LF	New line	0x2A	42	*	0x4A	74	J	0x6A	106	j
0x0B	11	VT	Vertical tab	0x2B	43	+	0x4B	75	K	0x6B	107	k
0x0C	12	FF	Form Feed	0x2C	44	,	0x4C	76	L	0x6C	108	l
0x0D	13	CR	Carriage return	0x2D	45	-	0x4D	77	M	0x6D	109	m
0x0E	14	SO	Shift out	0x2E	46	.	0x4E	78	N	0x6E	110	n
0x0F	15	SI	Shift in	0x2F	47	/	0x4F	79	O	0x6F	111	o
0x10	16	DLE	Data link escape	0x30	48	0	0x50	80	P	0x70	112	p
0x11	17	DC1	Device control 1	0x31	49	1	0x51	81	Q	0x71	113	q
0x12	18	DC2	Device control 2	0x32	50	2	0x52	82	R	0x72	114	r
0x13	19	DC3	Device control 3	0x33	51	3	0x53	83	S	0x73	115	s
0x14	20	DC4	Device control 4	0x34	52	4	0x54	84	T	0x74	116	t
0x15	21	NAK	Negative ack	0x35	53	5	0x55	85	U	0x75	117	u
0x16	22	SYN	Synchronous idle	0x36	54	6	0x56	86	V	0x76	118	v
0x17	23	ETB	End transmission block	0x37	55	7	0x57	87	W	0x77	119	w
0x18	24	CAN	Cancel	0x38	56	8	0x58	88	X	0x78	120	x
0x19	25	EM	End of medium	0x39	57	9	0x59	89	Y	0x79	121	y
0x1A	26	SUB	Substitute	0x3A	58	:	0x5A	90	Z	0x7A	122	z
0x1B	27	FSC	Escape	0x3B	59	;	0x5B	91	[	0x7B	123	{
0x1C	28	FS	File separator	0x3C	60	<	0x5C	92	\	0x7C	124	|
0x1D	29	GS	Group separator	0x3D	61	=	0x5D	93	]	0x7D	125	}
0x1E	30	RS	Record separator	0x3E	62	>	0x5E	94	^	0x7E	126	~
0x1F	31	US	Unit separator	0x3F	63	?	0x5F	95	_	0x7F	127	DEL

New Line Conventions

- UNIX / Linux: LF
- DOS / Windows: CR+LF
- Apple Mac (up to OS-9): CR

New Line Conventions

```
This is what happens^M
if you try to read a DOS/Windows file^M
on a UNIX/Linux machine!^M
```

New Line Conventions

```
This is what happens
                    if you try to read a
UNIX/Linux file
               on a Windows machine!
```

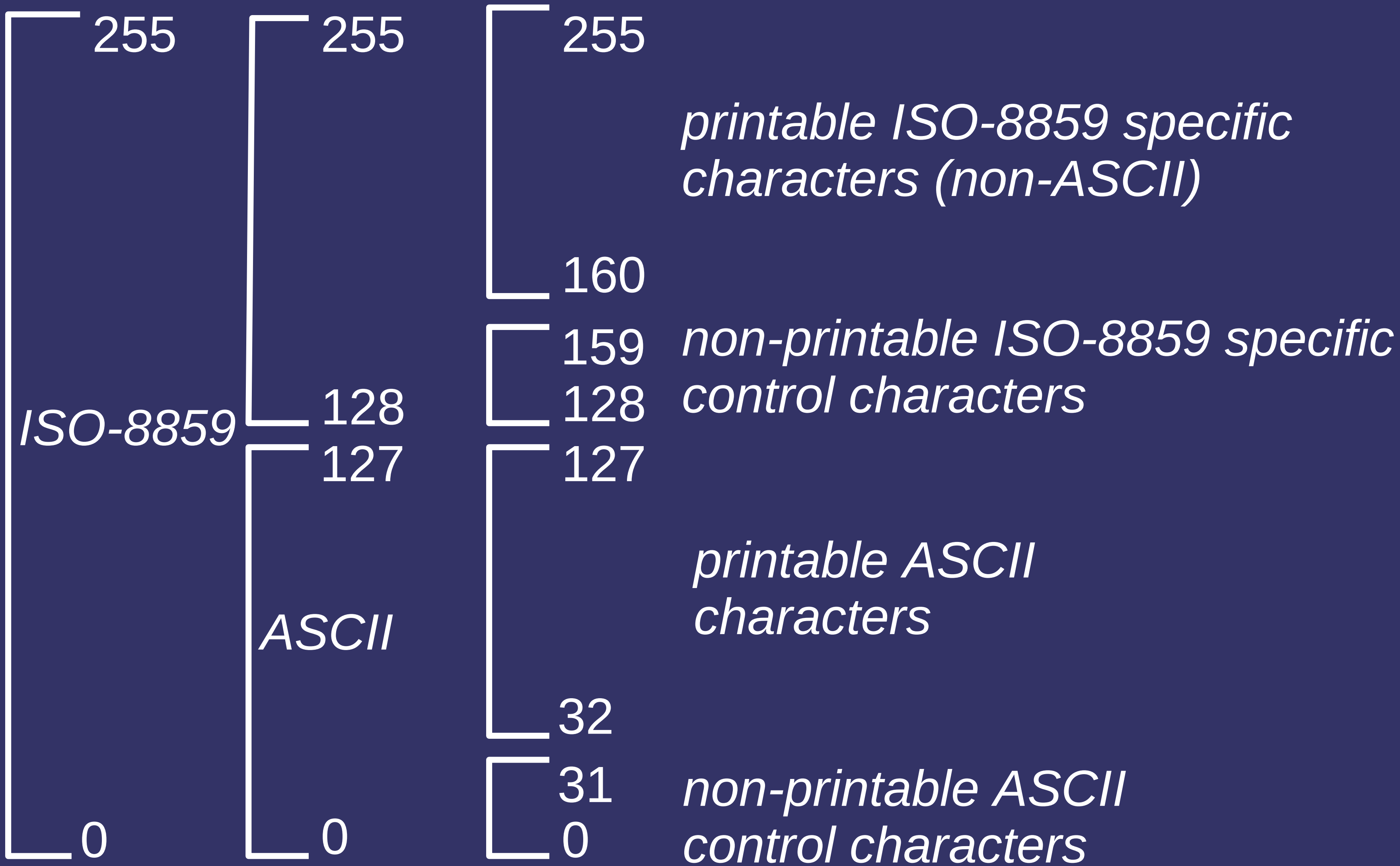
SOLUTION:
use Linux dos2unix/unix2dos/mac2unix/unix2mac tools
to convert from one new line convention to another
or use an editor than can handle each convention

8-bit Character Encoding: the ISO 8859 standards

- ASCII is a 7-bit code (128 characters only)
- ASCII does not support non-English characters
- For this, the ISO 8859 standards were invented
- Basic idea ISO 8859:
  - put a (language dependent) encoding “on top of” ASCII, using the full 8 bits (so 256 characters in total)
  - values 0-127 will yield the same characters as ASCII
  - values 128-255 will yield the additional characters needed for the particular non-English language

(values 0-31 and values 128-159
are non-printable control characters)

8-bit Character Encoding: the ISO 8859 standards



255

ISO-8859

0

255

128

127

ASCII

0

255

printable ISO-8859 specific characters (non-ASCII)

160

159

non-printable ISO-8859 specific control characters

128

127

printable ASCII characters

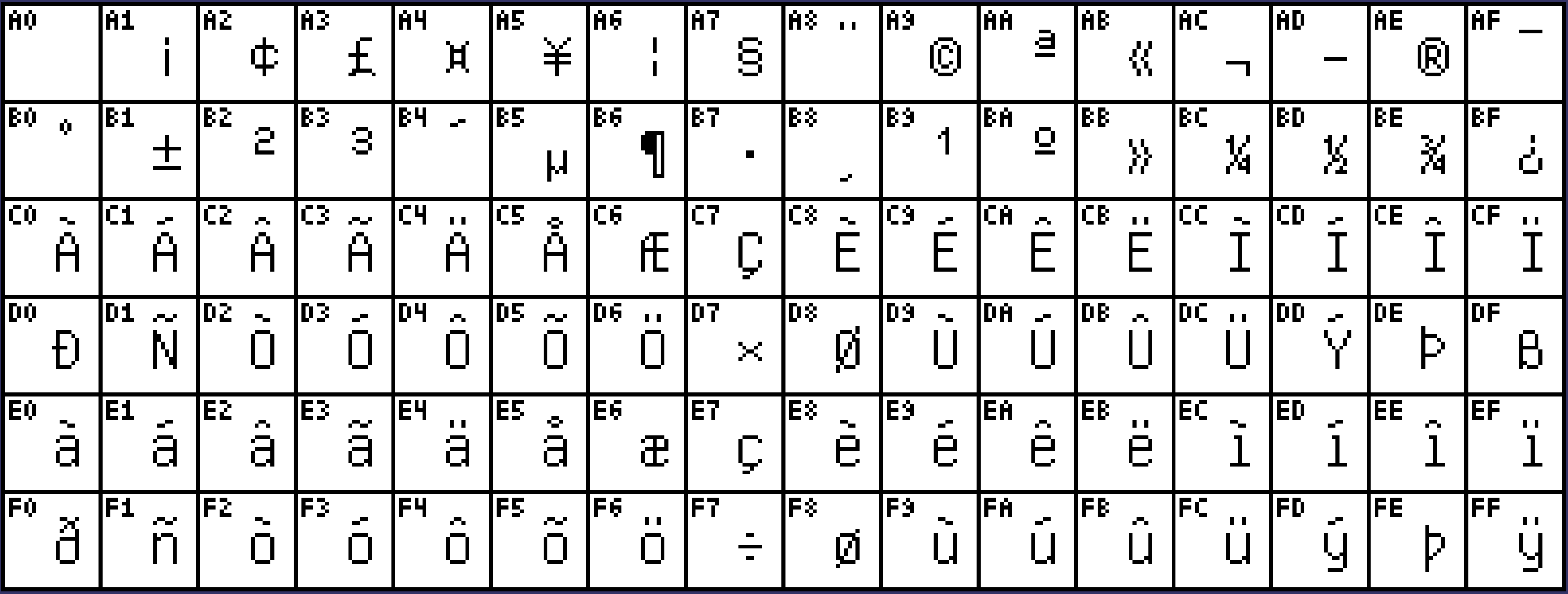
32

31

non-printable ASCII control characters

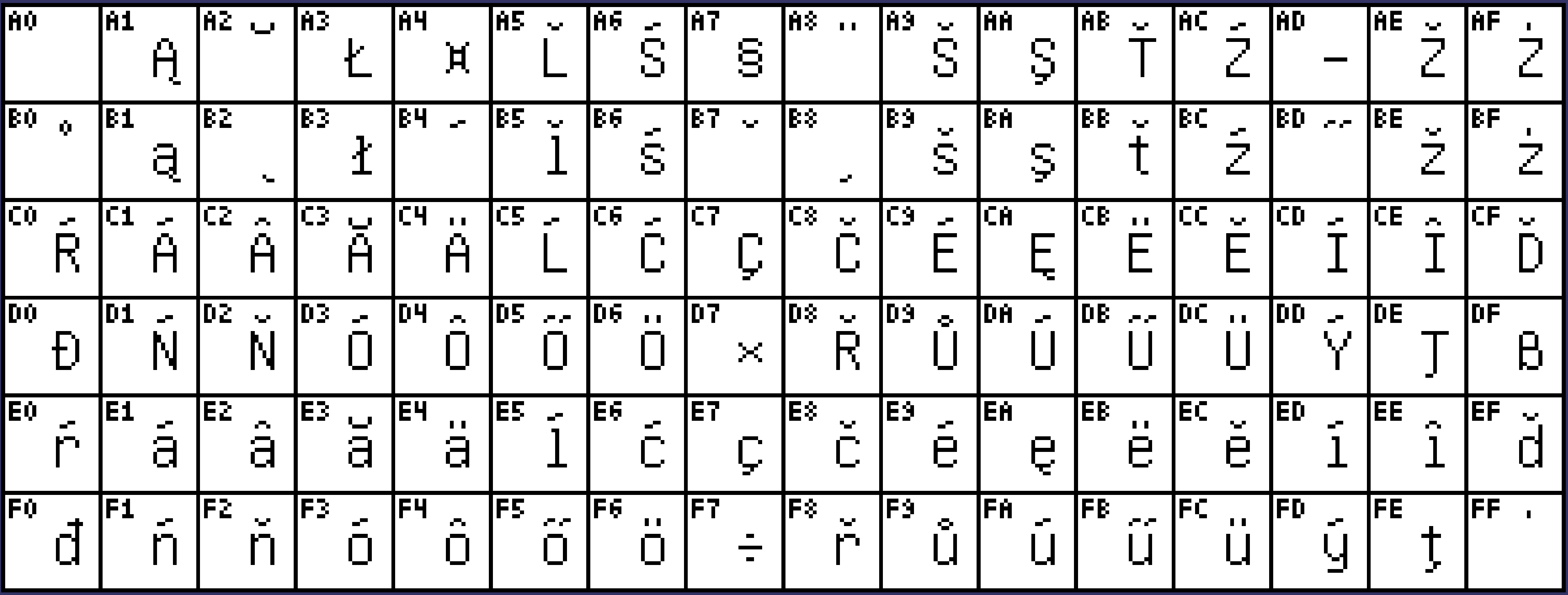
0

ISO 8859-1 / Latin-1
(Western Europe)



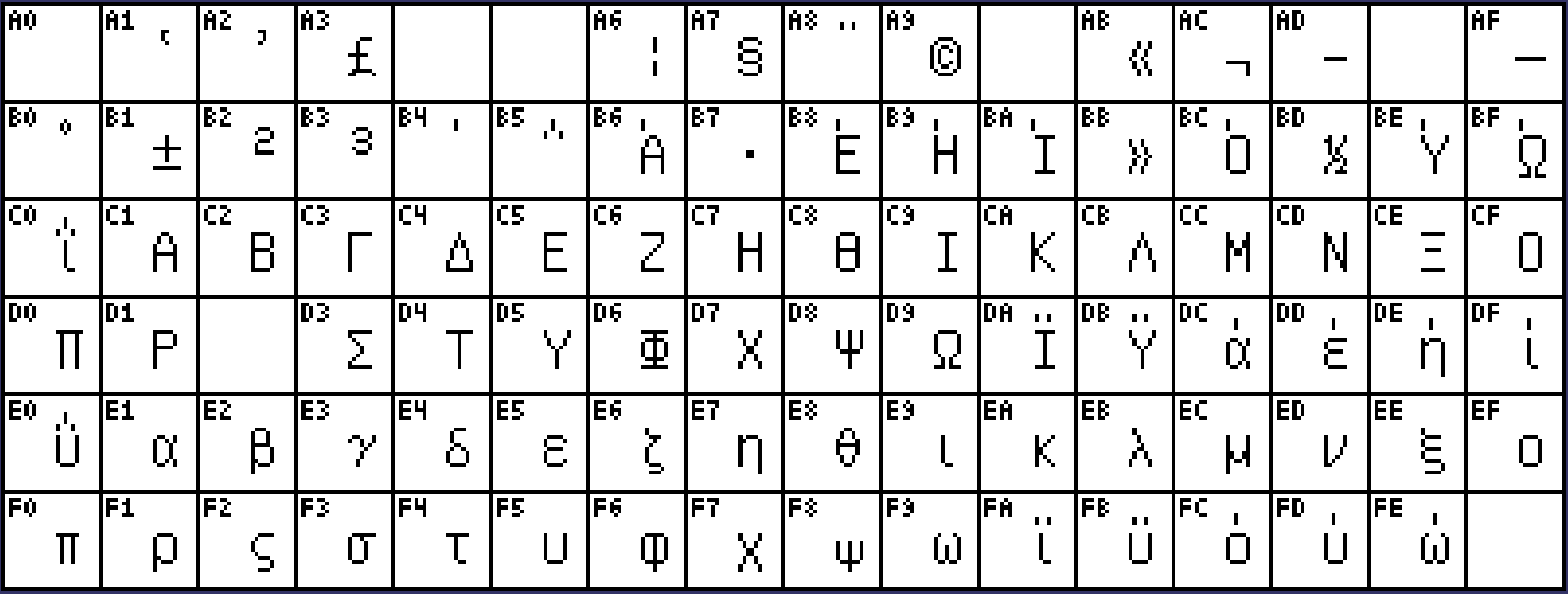
A0	A1 ¡	A2 ¢	A3 £	A4 ¤	A5 ¥	A6 ¦	A7 §	A8 ¨	A9 ©	AA ª	AB «	AC ¬	AD -	AE ®	AF ¯
B0 °	B1 ±	B2 ²	B3 ³	B4 ´	B5 µ	B6 ¶	B7 ·	B8 ¸	B9 ¹	BA º	BB »	BC ¼	BD ½	BE ¾	BF ¿
C0 À	C1 Á	C2 Â	C3 Ã	C4 Ä	C5 Å	C6 Æ	C7 Ç	C8 È	C9 É	CA Ê	CB Ë	CC Ì	CD Í	CE Î	CF Ï
D0 Ð	D1 Ñ	D2 Ò	D3 Ó	D4 Ô	D5 Õ	D6 Ö	D7 ×	D8 Ø	D9 Ù	DA Ú	DB Û	DC Ü	DD Ý	DE Þ	DF ß
E0 à	E1 á	E2 â	E3 ã	E4 ä	E5 å	E6 æ	E7 ç	E8 è	E9 é	EA ê	EB ë	EC ì	ED í	EE î	EF ï
F0 ð	F1 ñ	F2 ò	F3 ó	F4 ô	F5 õ	F6 ö	F7 ÷	F8 ø	F9 ù	FA ú	FB û	FC ü	FD ý	FE þ	FF ÿ

ISO 8859-2 / Latin-2 (Central Europe)



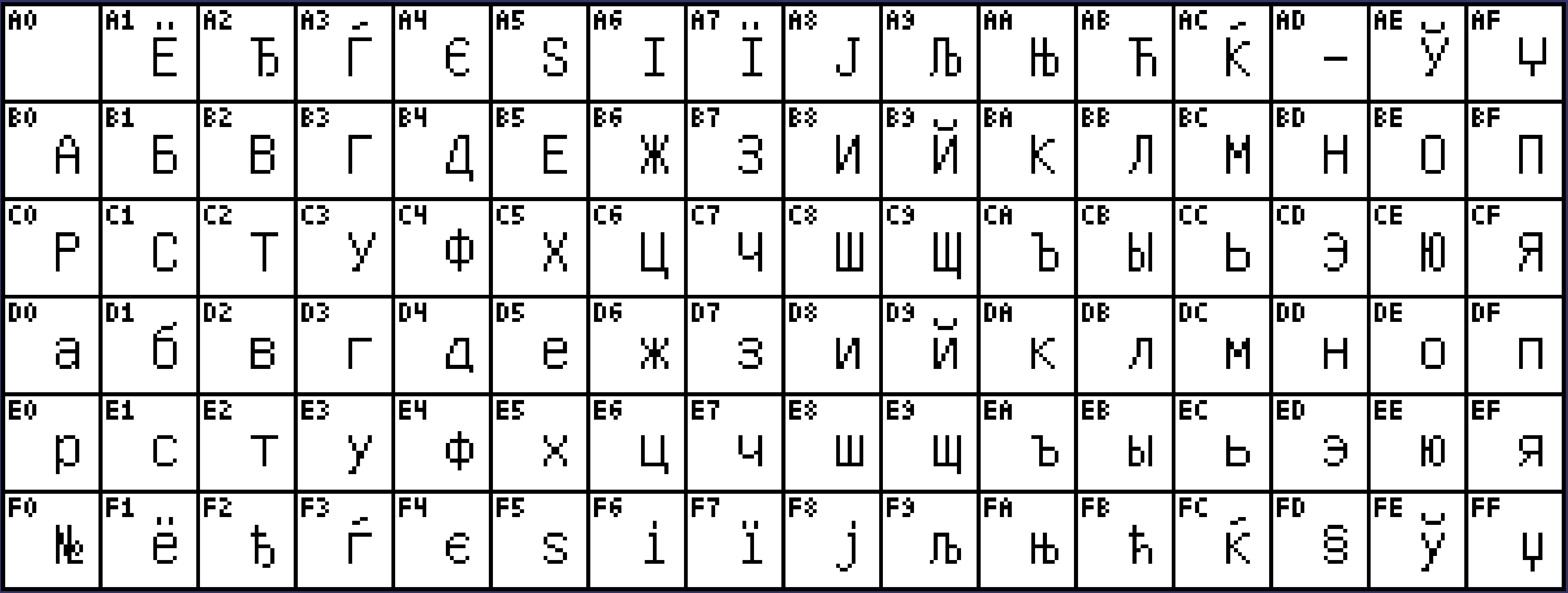
A0	A1 Ą	A2 ˘	A3 Ł	A4 ¤	A5 Ľ	A6 Ś	A7 §	A8 ¨	A9 Š	AA Ş	AB Ť	AC Ź	AD -	AE Ž	AF Ż
B0 °	B1 ą	B2 ˛	B3 ł	B4 ´	B5 ľ	B6 ś	B7 ˇ	B8 ¸	B9 š	BA ş	BB ť	BC ź	BD ˝	BE ž	BF ż
C0 Ŕ	C1 Á	C2 Â	C3 Ă	C4 Ä	C5 Ĺ	C6 Ć	C7 Ç	C8 Č	C9 É	CA Ę	CB Ë	CC Ě	CD Í	CE Î	CF Ď
D0 Đ	D1 Ń	D2 Ň	D3 Ó	D4 Ô	D5 Ő	D6 Ö	D7 ×	D8 Ř	D9 Ů	DA Ú	DB Ű	DC Ü	DD Ý	DE Ţ	DF ß
E0 ŕ	E1 á	E2 â	E3 ă	E4 ä	E5 ĺ	E6 ć	E7 ç	E8 č	E9 é	EA ę	EB ë	EC ě	ED í	EE î	EF ď
F0 đ	F1 ń	F2 ň	F3 ó	F4 ô	F5 ő	F6 ö	F7 ÷	F8 ř	F9 ů	FA ú	FB ű	FC ü	FD ý	FE ţ	FF ˙

ISO 8859-7
(Greek)



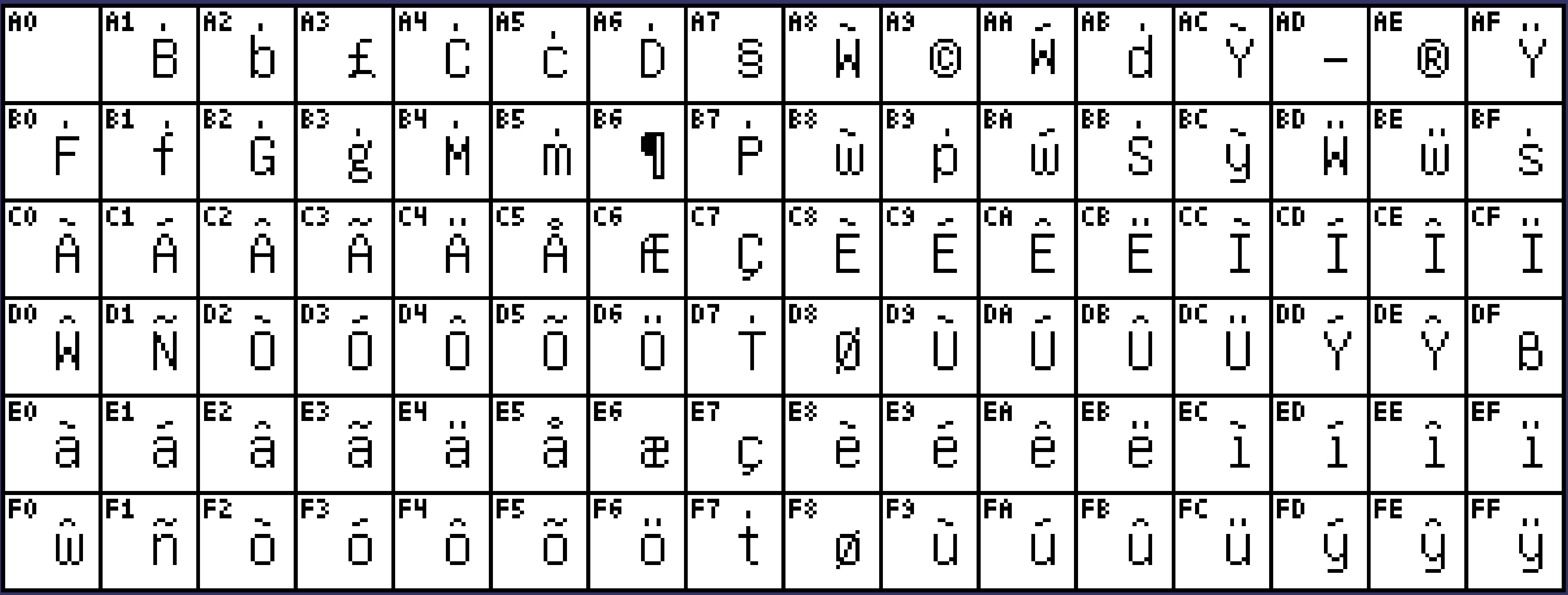
A0	A1 ‘	A2 ’	A3 £			A6 ¦	A7 §	A8 ¨	A9 ©		AB «	AC ¬	AD -		AF ―
B0 °	B1 ±	B2 ²	B3 ³	B4 ΄	B5 ΅	B6 Ά	B7 ·	B8 Έ	B9 Ή	BA Ί	BB »	BC Ό	BD ½	BE Ύ	BF Ώ
C0 ΐ	C1 Α	C2 Β	C3 Γ	C4 Δ	C5 Ε	C6 Ζ	C7 Η	C8 Θ	C9 Ι	CA Κ	CB Λ	CC Μ	CD Ν	CE Ξ	CF Ο
D0 Π	D1 Ρ		D3 Σ	D4 Τ	D5 Υ	D6 Φ	D7 Χ	D8 Ψ	D9 Ω	DA Ϊ	DB Ϋ	DC ά	DD έ	DE ή	DF ί
E0 ΰ	E1 α	E2 β	E3 γ	E4 δ	E5 ε	E6 ζ	E7 η	E8 θ	E9 ι	EA κ	EB λ	EC μ	ED ν	EE ξ	EF ο
F0 π	F1 ρ	F2 ς	F3 σ	F4 τ	F5 υ	F6 φ	F7 χ	F8 ψ	F9 ω	FA ϊ	FB ϋ	FC ό	FD ύ	FE ώ	

ISO 8859-5
(Cyrillic)



A0	A1 Ё	A2 Ђ	A3 Ѓ	A4 Є	A5 Ѕ	A6 І	A7 Ї	A8 Ј	A9 Љ	AA Њ	AB Ћ	AC Ќ	AD -	AE Ў	AF Џ
B0 А	B1 Б	B2 В	B3 Г	B4 Д	B5 Е	B6 Ж	B7 З	B8 И	B9 Й	BA К	BB Л	BC М	BD Н	BE О	BF П
C0 Р	C1 С	C2 Т	C3 У	C4 Ф	C5 Х	C6 Ц	C7 Ч	C8 Ш	C9 Щ	CA Ъ	CB Ы	CC Ь	CD Э	CE Ю	CF Я
D0 а	D1 б	D2 в	D3 г	D4 д	D5 е	D6 ж	D7 з	D8 и	D9 й	DA к	DB л	DC м	DD н	DE о	DF п
E0 р	E1 с	E2 т	E3 у	E4 ф	E5 х	E6 ц	E7 ч	E8 ш	E9 щ	EA ъ	EB ы	EC ь	ED э	EE ю	EF я
F0 №	F1 ё	F2 ђ	F3 ѓ	F4 є	F5 ѕ	F6 і	F7 ї	F8 ј	F9 љ	FA њ	FB ћ	FC ќ	FD §	FE ў	FF џ

ISO 8859-14 / Latin-8
(Welsh, Cornish, Gaellic, Irish, ...)



A0	A1 Ḃ	A2 ḃ	A3 £	A4 Ċ	A5 ċ	A6 Ḋ	A7 §	A8 Ẁ	A9 ©	AA Ẃ	AB ḋ	AC Ỳ	AD -	AE ®	AF Ÿ
B0 Ḟ	B1 ḟ	B2 Ġ	B3 ġ	B4 Ṁ	B5 ṁ	B6 ¶	B7 Ṗ	B8 ẁ	B9 ṗ	BA ẃ	BB Ṡ	BC ỳ	BD Ẅ	BE ẅ	BF ṡ
C0 À	C1 Á	C2 Â	C3 Ã	C4 Ä	C5 Å	C6 Æ	C7 Ç	C8 È	C9 É	CA Ê	CB Ë	CC Ì	CD Í	CE Î	CF Ï
D0 Ŵ	D1 Ñ	D2 Ò	D3 Ó	D4 Ô	D5 Õ	D6 Ö	D7 Ṫ	D8 Ø	D9 Ù	DA Ú	DB Û	DC Ü	DD Ý	DE Ŷ	DF ß
E0 à	E1 á	E2 â	E3 ã	E4 ä	E5 å	E6 æ	E7 ç	E8 è	E9 é	EA ê	EB ë	EC ì	ED í	EE î	EF ï
F0 ŵ	F1 ñ	F2 ò	F3 ó	F4 ô	F5 õ	F6 ö	F7 ṫ	F8 ø	F9 ù	FA ú	FB û	FC ü	FD ý	FE ŷ	FF ÿ

Roundup ISO 8859 Character Encodings

- advantages:
  - does not require any additional space (ASCII doesn't use the 8th bit anyway)
  - relative simplicity (once you know the code page)

- disadvantages:
  - what if the same page needs several languages?
  - what about languages with more than 128 special characters (Chinese, Japanese, ...)

Unicode

- assigns to each character
  a unique number (“*code point*”)

A:	U+0041
£:	U+00A3
α:	U+03B1
女 :	U+F981

- numbers 0-255 correspond
  with ISO 8859-1 character set
  (which includes ASCII)

- Unicode by itself doesn't say anything
  about how things are encoded at byte level!

Encoding Unicode at Byte Level

- UCS-2: just use 2 bytes for each code point
  (instead of 1 just for ASCII/ISO-8859)
  Disadvantages:
    - it's not backward compatible with ASCII
    - Unicode now has more than 65t code points
    - it's generally considered obsolete (don't use it!)

- UTF-8: use 1 byte if it's an ASCII character and multiple bytes if it's not (using a clever way of encoding that also specifies the length of multiple byte characters)
  Advantages:
    - it's backward compatible with ASCII
    - can handle *all* Unicode code points
    - it's becoming the standard on the Web

UTF-8 technical details

number of bits	first code point	last code point	byte 1	byte 2	byte 3	byte 4
0-7	U+0000	U+007F	**0**xxxxxxx			
8-11	U+0080	U+07FF	**110**xxxxx	**10**xxxxxx		
12-16	U+0800	U+FFFF	**1110**xxxx	**10**xxxxxx	**10**xxxxxx	
16-21	U+10000	U+10FFF	**11110**xxx	**10**xxxxxx	**10**xxxxxx	**10**xxxxxx

Please note that:

- byte 1 indicates how many bytes follow
- any UTF-8 byte can be identified as a start byte or follow-up byte
- UTF-8 is compatible to ASCII (why?)
- UTF-8 is *not* backwards compatible with ISO-8859-1 (why?)

UTF-8 versus ISO-8859-1

What you entered: `welcome to Lancôme`

What is displayed: `welcome to LancÃ´me`

Can you see what is going on?

ô = U+C3 = 11110100

UTF-8 encoding:

11000011 **10**110100

Ã ´

(ISO 8859-1 interpretation)

UTF-8 versus ISO-8859-1

What you entered: `welcome to Lancôme`

What is displayed: `welcome to Lancme`

Can you see what is going on?

ô = U+C3 = 11110100

m = U+6D = 01101101

```
11110100   01101101
      error   m
```

(UTF-8 interpretation)

Take Home Message

- Unicode with UTF-8 is usually the safe option (recommended as default encoding by W3C)
- If you're writing your pages in just a single European language, using an ISO 8859 encoding will give you a small efficiency gain (each character is just 1 byte)
- If you're planning to use just ASCII characters, it doesn't matter whether you're using ISO 8859 or UTF-8 because it's all the same!
- Make sure your editor saves your file in the right format!

How to Recognize the Character Encoding

1) Guessing, based on a statistical analysis of the file contents (not recommended)

2) “Byte Order Mark” at the beginning of the file (like *EF BB BF* for UTF-8) (not recommended)

3) In the HTTP header:
   Content-Type: text/html; charset=utf-8
   (or *us-ascii*, *iso-8859-1*, *iso-8859-2*, etc.)
   You'd need to configure your web server to do this.

Example of Character Encoding in HTTP Header

```
GET / HTTP/1.1
Host: www.cs.cf.ac.uk
```

this is what the browser would send (simplified)

```
HTTP/1.1 200 OK
Date: Wed, 28 Oct 2015 17:39:21 GMT
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.3.3
Connection: close
Content-Type: text/html; charset=UTF-8
```

this is what the web server would reply (HTTP header, simplified)

```
<html>
<head>
  <title>An Example Page</title>
</head>
<body>
  <p>Hello World!<br>How are you?</p>
</body>
</html>
```

after sending the HTTP header, the web server sends the actual HTML file

How to Recognize the Character Encoding

1) Guessing, based on a statistical analysis of the file contents (not recommended)

2) “Byte Order Mark” at the beginning of the file (like *EF BB BF* for UTF-8) (not recommended)

3) In the HTTP header:
   Content-Type: text/html; charset=utf-8
   (or *us-ascii*, *iso-8859-1*, *iso-8859-2*, etc.)
   You'd need to configure your web server to do this.

4) In the HTML file itself:
   <meta charset=”utf-8”>
   (or *us-ascii*, *iso-8859-1*, *iso-8859-2*, etc.)

Example of Character Encoding in HTML file

```
<html>
<head>
  <meta charset="utf8">
  <title>An Example Page</title>
</head>
<body>
  <p>Hello World!<br>How are you?</p>
</body>
</html>
```

What Plain Text Files Do Not Encode

A plain text file (be it ASCII, Latin-1 or Unicode/UTF8) does not encode:

- any particular font (Times, Arial, etc.)
- any particular font size (11pt, 12pt, etc.)
- any special formatting (*italics*, **bold**, underline, etc.)
- any particular colouring scheme

Word processors use more advanced file formats that can store these, but these formats are not plain text.

HTML requires plain text; this is why you cannot use MS Word to write HTML (unless you really know what you're doing). Use a plain text editor (*Sublime* or *vi*) instead!

How HTML Exceeds the Limitations of Plain Text

- Question: If HTML uses plain text, then how can browsers display any special formatting?
- Answer: Because of *markup.*

```
HTML uses markup tags to indicate structure
or special formatting. <i>This text is
displayed in italics</i> whereas <b>this
text is displayed bold.</b>
```

HTML uses markup tags to indicate structure or special formatting. *This text is displayed in italics* whereas **this text is displayed bold.**

How HTML Exceeds the Limitations of Plain Text

- Question: If HTML uses plain text, then how can browsers display any special formatting?
- Answer: Because of *markup.*

```
HTML uses markup tags to indicate structure
or special formatting. <em>This text is to
be emphasized</em> whereas <strong>this text
is to be strongly emphasized.</strong>
```

HTML uses markup tags to indicate structure or special formatting. *This text is to be emphasized* whereas **this text is to be strongly emphasized.**

An Example of HTML

```
<!DOCTYPE html>

<html>

<head>
  <meta charset=”utf-8”/>
  <title>An Example Page</title>
</head>

<body>
  <p>Hello world!<br/>How are you?</p>
</body>

</html>
```

Some Key Concepts of HTML

- tags:
  ```
  <html>, </html>, <title>, </title>, <br>, …
  ```
- attributes/values:
  ```
  <meta charset=”utf-8”>
  ```
- elements:
  ```
  <title>An Example Page</title>
  ```
- nested elements:
  ```
  <body><p>Hello World!</p></body>
  ```
- empty elements:
  ```
  <br/>
  <meta charset=”utf-8”/>
  ```