A page addressed by one-byte codes

Every letter on the three lines below costs one byte of this file's page stream instead of two. A font is reached by glyph index unless it is told otherwise, two bytes a glyph, which puts every glyph the face holds within reach. Reached through an encoding it is addressed by one-byte codes instead: at most 255 of them, each standing for one character, and the file states which character each of them stands for.

Set through one-byte codes

HQF Development

An invoice, a payslip, a delivery note.

One byte a letter, and the file says which.

The codes it settled

The encoding is fitted to the words this page draws: the space keeps code 32, and every other character takes the lowest code free from 33 up, so nothing in the stream is a byte reading software has to escape. The first of them are H 33 · Q 34 · F 35 · D 36 · e 37 · v 38 · l 39 · o 40.

What the stream weighs

Drawn by glyph index, the three lines above cost 526 bytes of page stream; drawn through one-byte codes, 332 bytes, which is 194 less. Both figures are measured by the program that drew this page, not written in by hand.

What the single byte 32 buys

Words reading software can push apart.

Words reading software can push apart.

The two lines above hold the same characters. The second is drawn with word spacing set, which moves the pen at the single byte 32 and at no other code. Under glyph indices no code is a single byte 32, so the operator has nothing to reach and the words never move; the space of this encoding sits at 32, so they do.

A code page named by its standard

Счёт-фактура № 117 — оплата до 30 сентября

The line above is set through Windows code page 1251, named once rather than spelled out code by code: the file states the Windows encoding and names over it every code the page fills differently, so a reader finds each Cyrillic letter where the page puts it. Its first bytes are D1 F7 B8 F2 2D F4 E0 EA.

One byte addresses 255 characters at a time, so a page of several alphabets is still reached by glyph index. No ligature and no small capital is drawn through an encoding either: those are glyphs no character stands for, and a code stands for a character. A file that claims archival conformance states the Windows encoding as the one its codes are read against, which this one does.

Thirty-one encodings a code can be read against

What the byte 0xC9 stands for in each: the character's code point, and the character itself where this face can draw it.

WinAnsi	U+00C9 É	ISO 8859-9	U+00C9 É	Windows-1254	U+00C9 É
Standard	nothing	ISO 8859-10	U+00C9 É	Windows-1255	U+05B9 ֹ
PDFDoc	U+00C9 É	ISO 8859-11	U+0E29	Windows-1256	U+0629 ة
ISO 8859-1	U+00C9 É	ISO 8859-13	U+00C9 É	Windows-1257	U+00C9 É
ISO 8859-2	U+00C9 É	ISO 8859-14	U+00C9 É	Windows-1258	U+00C9 É
ISO 8859-3	U+00C9 É	ISO 8859-15	U+00C9 É	Mac Roman	U+2026 …
ISO 8859-4	U+00C9 É	ISO 8859-16	U+00C9 É	IBM 037	U+0049 I
ISO 8859-5	U+0429 Щ	Windows-874	U+0E29	IBM 500	U+0049 I
ISO 8859-6	U+0629 ة	Windows-1250	U+00C9 É	Mac Expert	U+F7B8
ISO 8859-7	U+0399 Ι	Windows-1251	U+0419 Й		
ISO 8859-8	nothing	Windows-1253	U+0399 Ι		

An encoding settled by hand

code 0x41 stands for U+0416, whose glyph is named Zhecyrillic
code 0x42 stands for U+03A9, whose glyph is named Omegagreek
code 0x43 stands for U+20AC, whose glyph is named Euro
code 0x44 stands for U+00C9, whose glyph is named Eacute

ЖΩ€É

The line above is drawn through those four codes alone. The encoding starts from no encoding at all, and the file names 5 codes that differ from it.

A face that lacks the ligature U+FB01 is handed “fi” in its place, and one that lacks the Ohm sign U+2126, “Ω”.

What a face draws for a character it cannot draw is its glyph 0, which every subset keeps.