

2

T7: UNICODE

Unicode → 3 Unicode Transformation Format (UTF)

- Acceptar código Unicode no Bytes (menos los surrogates)

UTF-8
UTF-16
UTF-32

UTF-8

Transforma Unicode a 1, 2, 3 o 4 Bytes

Binary Format and Split Bytes				
Code Point Range	Byte 1	Byte 2	Byte 3	Byte 4
U+000000... U+00007F...	aaaaaaa	aaaaaaa		
U+000080... U+0000FF...	bbbbbbb	aaaaaaa		
U+000080... U+0000FF...	11bbbbb	10aaaaaa		
U+000080... U+0000FF...	110cccc	10bbbbbb	10aaaaaa	
U+000080... U+0000FF...	ddcccccc	bbbbbbaaaaa		
U+10FFFF...	1110ddd	10cccccc	10bbbbbb	10aaaaaa

Ventajas

- código ASCII = código formateado en utf-8
- codePoint Mayor = código utf-8 nuevos (se puede ordenar)
- Inmune a los problemas de endianess

UTF-32

Allo todos los codepoints en 32 bits o tomar por culo 4 bytes

Ventajas

- Más sencillo
- El codepoint 10 veces más rápida
- Random Access: se donde empieza quiero ir al 5º carácter = 4 bytes

¿Cuando se usa? → en Memoria → se usa a espacio vs rapidez

Encodings

Guardan todo el juego de caracteres en 1 formato

UTF-16

lo guarda en 2 Bytes

→ 0+0000 = U+00FF
2
+ 0+0000 = U+00FF

+ U+1000 = U+10FF
se usa un rango de 2x16bits

* Rangos de surrogates

Quieres codificar emojis → Su unicode no se puede representar en solo 16bits

Tengo unos caracteres "surrogates" o "substitutes" q. no se usan

Los emojis: los codificaron una pareja de surrogates

• 0x010000 → se divide del codepoint, 16 q. dejo, un rango de 20 bits: 0...0XFFFF

"High surrogate" 10 bits + 10 bits → "Low surrogate"
Rango: 0XDC00 - 0XDBFF Rango: 0XDC00 - 0XDC00

→ U+D800 + U+0FFF → UTF-16 no los puede codificar (caracteres ilegales)
"Surrogates"

Ventajas

- CJK → en UTF-8 usan 3-4 bytes, en UTF-16 2-3

Problemas

- Endianess
- Necesidad de surrogates
- Desaprovechamiento espacial → ASCII, ...

SOLUCIONAR PROBLEMAS ENDIANESS

Byte Ordering Mask



por defecto unicode usa Big Endian pero para estos rangos usan unicode

Antes de empesar el archivo tienen una secuencia: FE FF



UTF-16



UTF 32 → 0000 FEFF

UTF 8 → Endianess no te afecta

Ejemplo Encoding unicode

UTF-32 = 10331 (one 32-bit value / code point)

UTF-16 = D800 DF31 (one or two 16-bit values / code point)

UTF-8 = F0 90 8C B1 (one to four 8-bit values / code point)

UTF-16 Surrogates: D800-DFFF

High: D800-DBFF, Low: DC00-DFFF



Surrogates used to access 10000-10FFFF in UTF-16