# If You Liked It Then You Should Have Put A String On It

360|iDev 2019

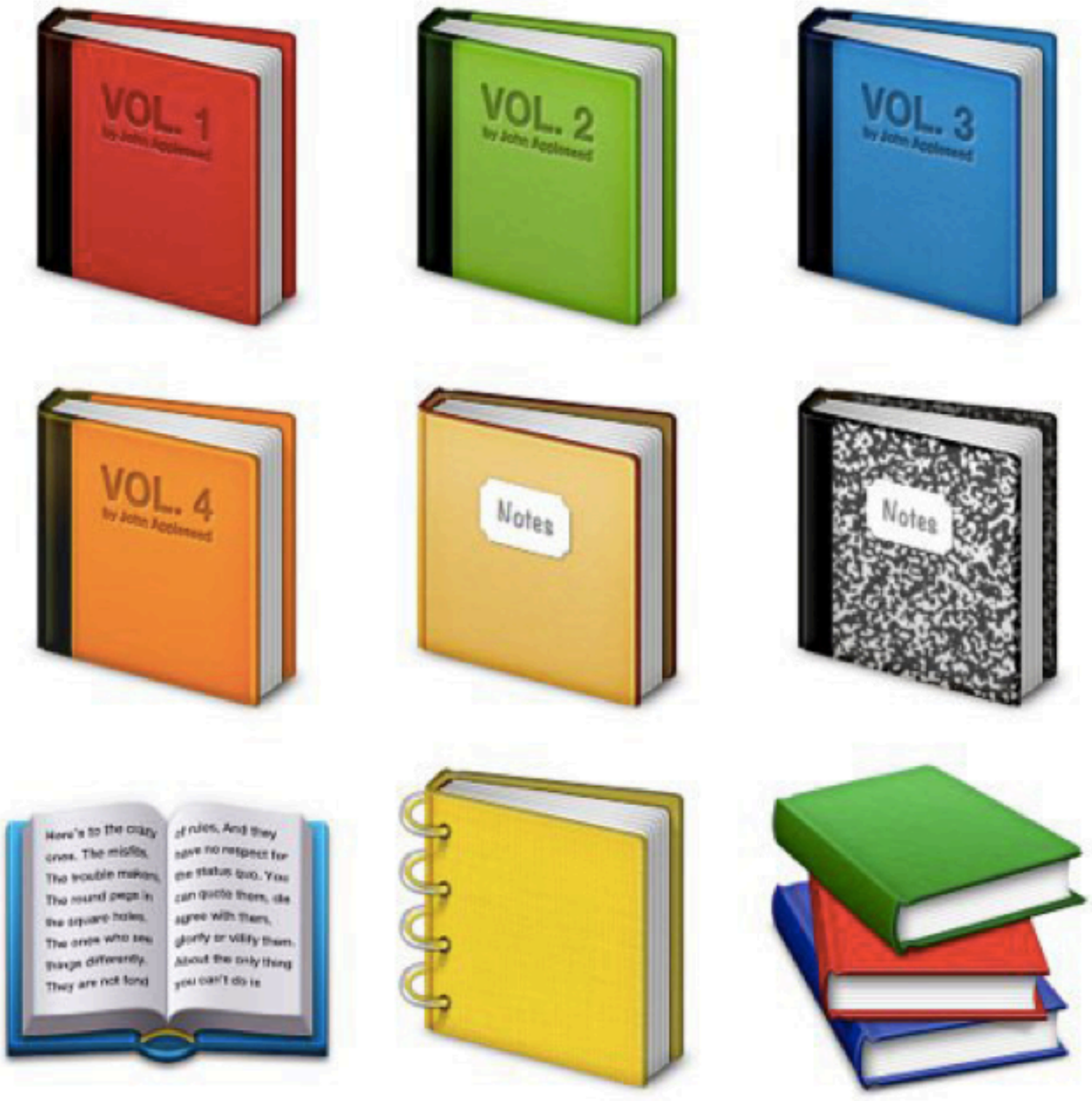
Rachel Hyman

@shibasprinkle

github.com/rachelhyman/emoji

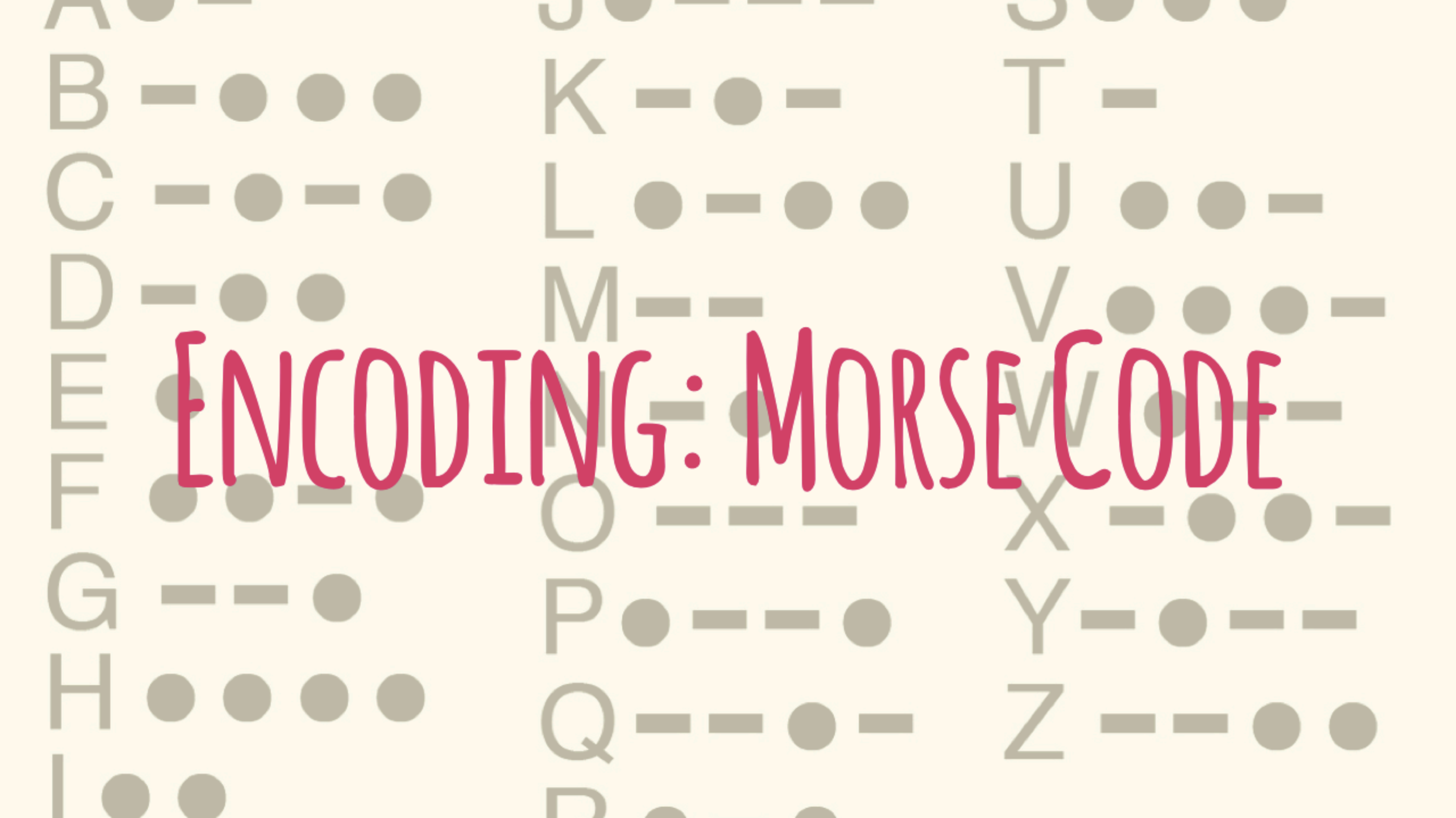
venmo

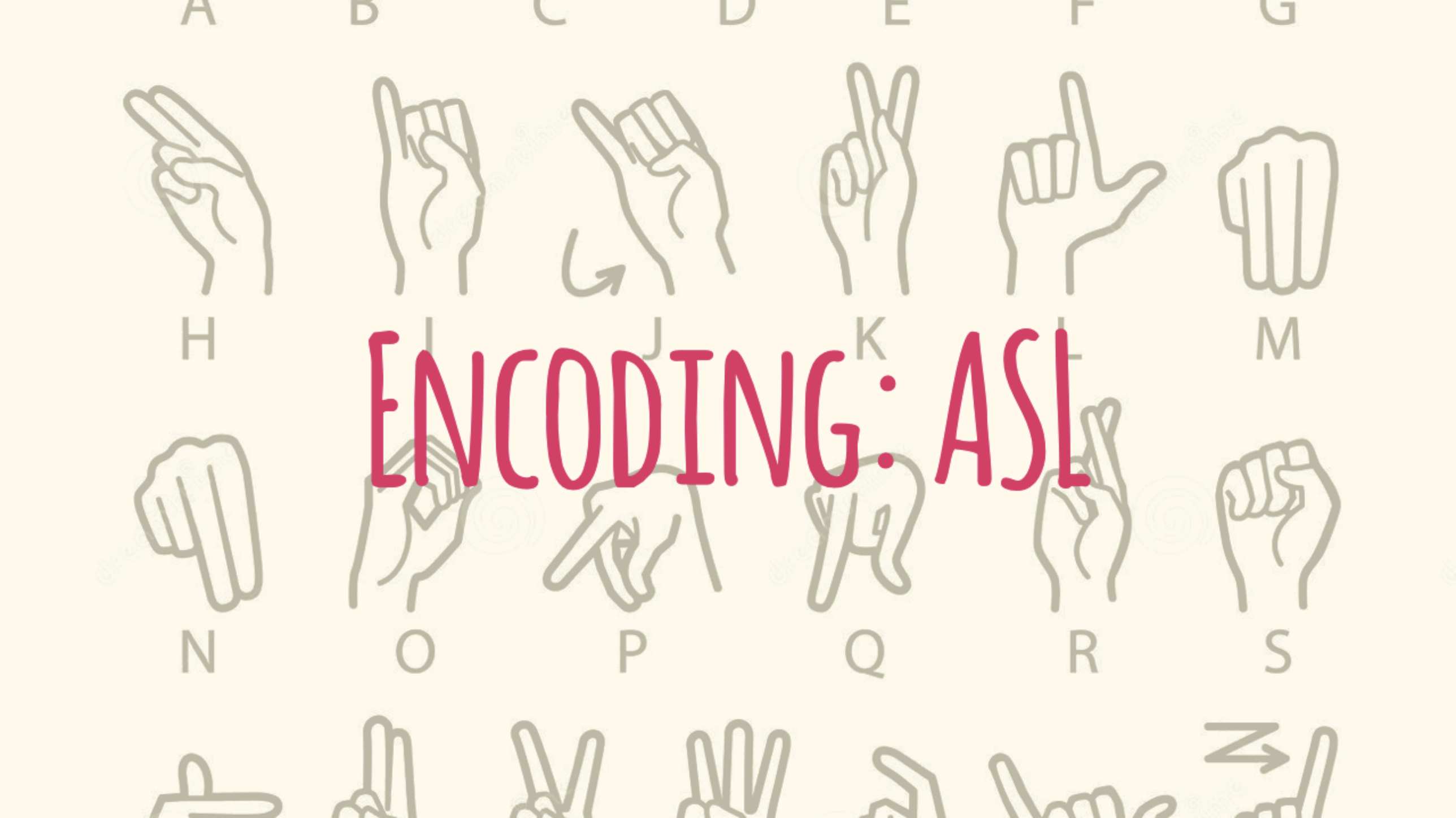
# Italic—Type



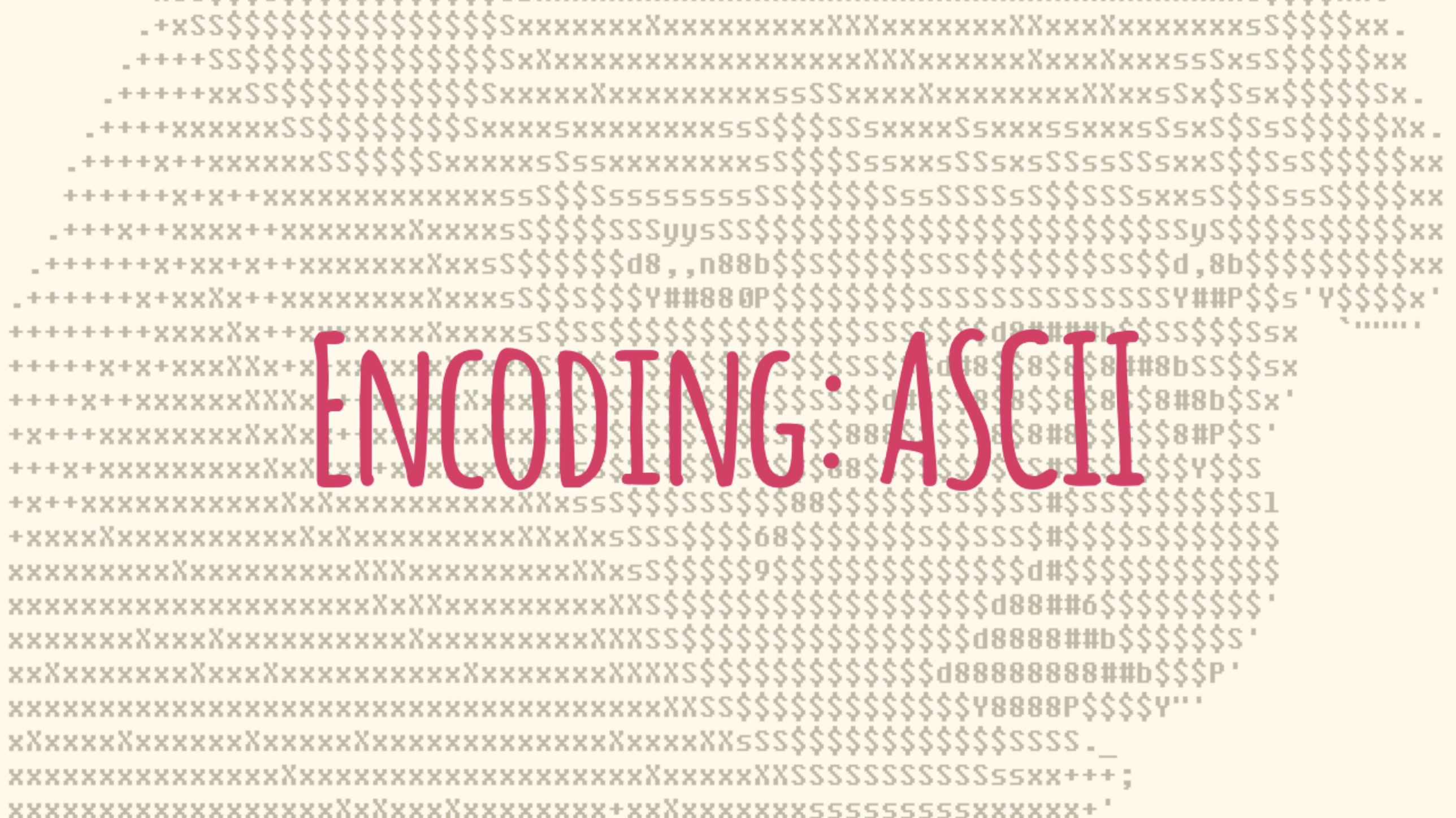
An emoji is
a type of string.

# Encoding


ENCODING: MORSE CODE




# ENCODING: ASL


ENCODING: ASCII

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | NUL | DLE | space | 0 | @ | P | ` | p |
| 1 | SOH | DC1<br>XON | ! | 1 | A | Q | a | q |
| 2 | STX | DC2 | " | 2 | B | R | b | r |
| 3 | ETX | DC3<br>XOFF | # | 3 | C | S | c | s |
| 4 | EOT | DC4 | $ | 4 | D | T | d | t |
| 5 | ENQ | NAK | % | 5 | E | U | e | u |
| 6 | ACK | SYN | & | 6 | F | V | f | v |
| 7 | BEL | ETB | ' | 7 | G | W | g | w |
| 8 | BS | CAN | ( | 8 | H | X | h | x |
| 9 | HT | EM | ) | 9 | I | Y | i | y |
| A | LF | SUB | * | : | J | Z | j | z |
| B | VT | ESC | + | ; | K | [ | k | { |
| C | FF | FS | , | < | L | \ | l | \| |
| D | CR | GS | - | = | M | ] | m | } |
| E | SO | RS | . | > | N | ^ | n | ~ |
| F | SI | US | / | ? | O | _ | o | del |

Stored data:

| ... | 66 | 105 | 108 | 108 | 32 | 115 | 108 | 111 | 119 | 108 | 121 | 32 | 116 | 117 | 114 | 110 | 101 | 100 | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

How the data is interpreted according to ASCII:

... B i l l s l o w l y t u r n e d ...

How the data might be interpreted according to another encoding:

... ן ד ח ח ן ח ש ל ח א ם ת ר ב ע ה ...

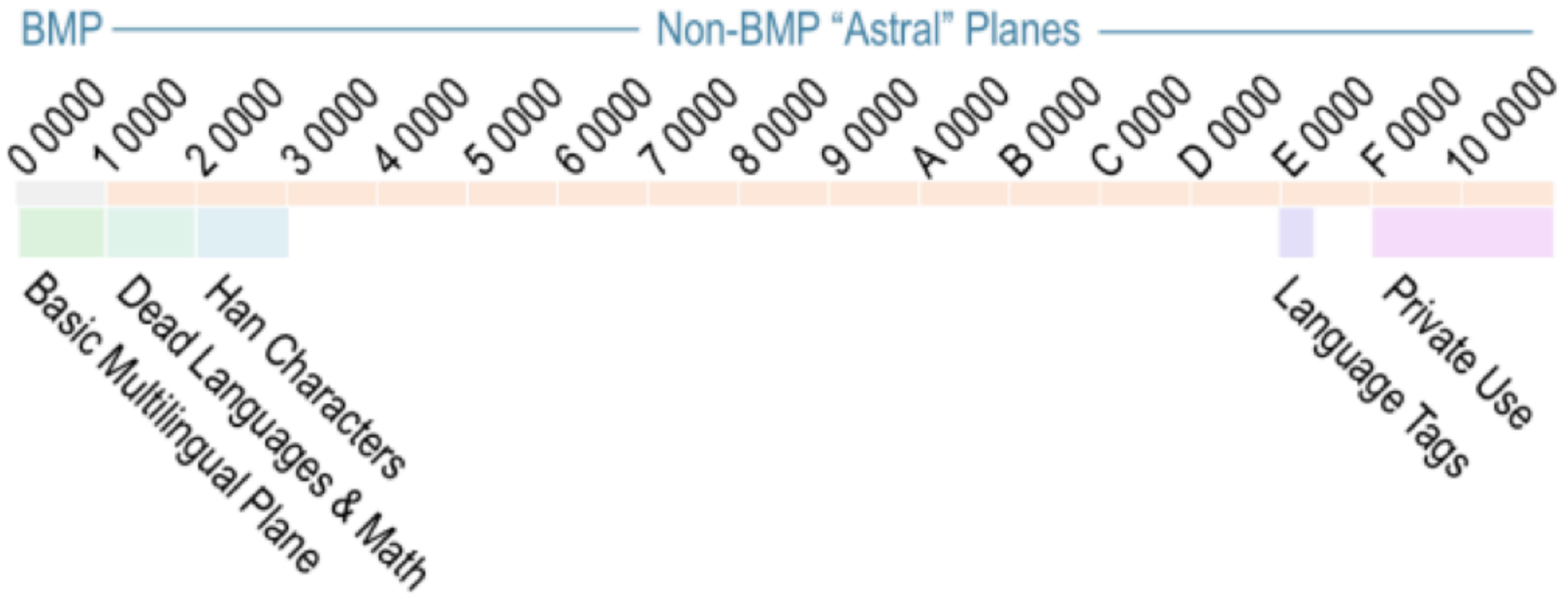

# ENCODING: UNICODE

"Unicode provides a unique number for every character, no matter what the platform, program, or language is."

-Unicode

# 1,114,112 Unicode Code Points

## 17 “Planes” each with 64k code points: U+0000 – U+10FFFF

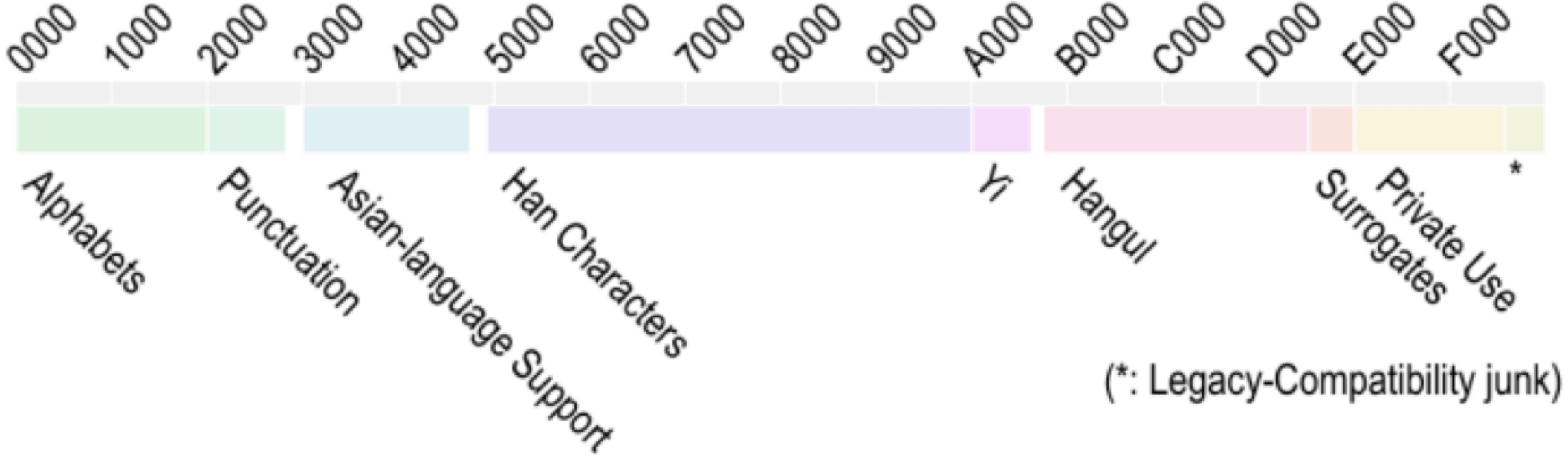



~137,000

characters defined in Unicode 12.1

# The Basic Multilingual Plane (BMP)

## U+0000 – U+FFFF

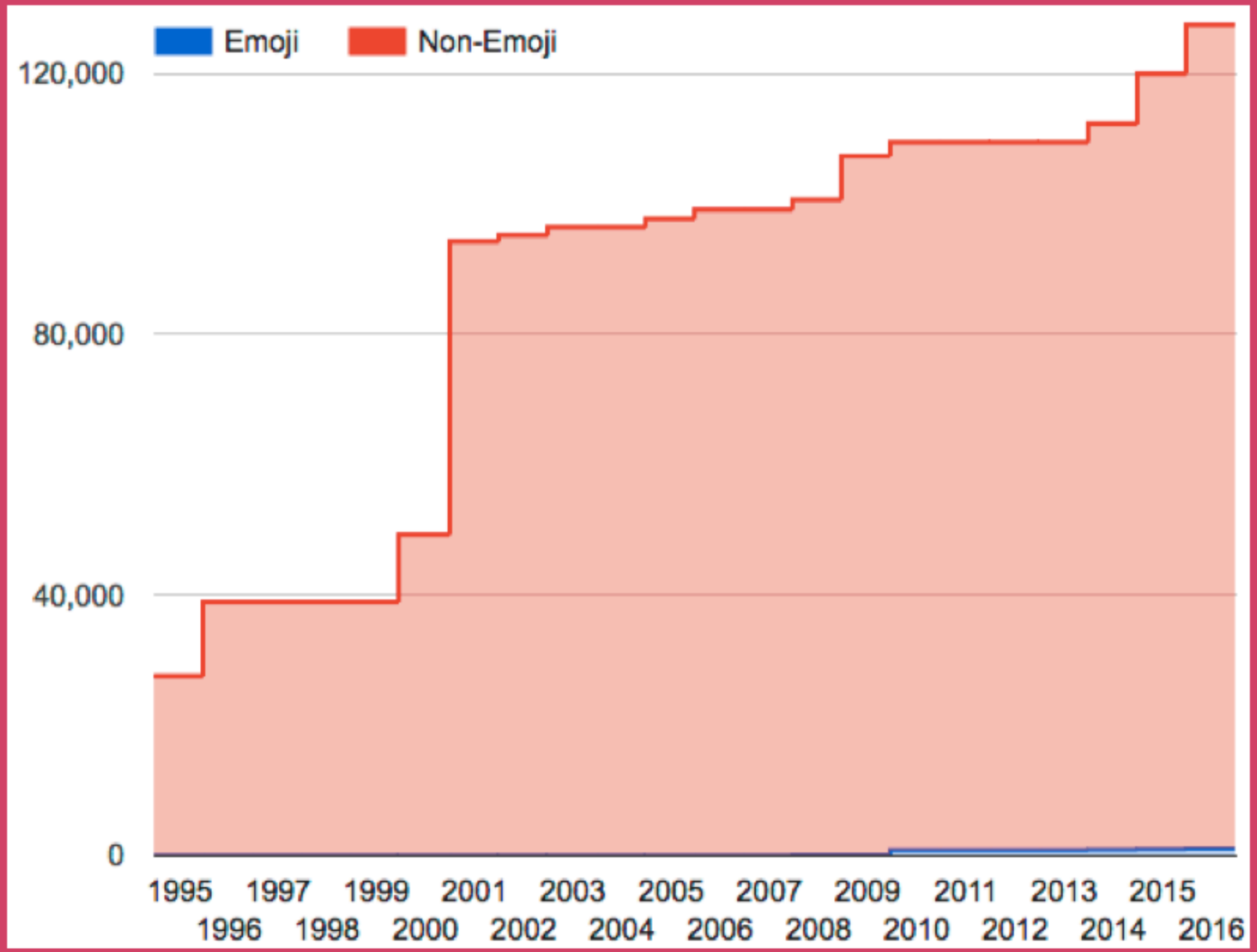



(*: Legacy-Compatibility junk)

| African Scripts |
| --- |
| ***Adlam*** |
| ***Bamum*** |
| Bamum Supplement |
| ***Bassa Vah*** |
| ***Coptic*** |
| *Coptic in Greek block* |
| Coptic Epact Numbers |
| ***Egyptian Hieroglyphs (1MB)*** |
| Egyptian Hieroglyph Format Controls |
| ***Ethiopic*** |
| Ethiopic Supplement |
| Ethiopic Extended |
| Ethiopic Extended-A |
| ***Medefaidrin*** |
| ***Mende Kikakui*** |
| ***Meroitic*** |
| Meroitic Cursive |
| Meroitic Hieroglyphs |
| ***N'Ko*** |
| ***Osmanya*** |
| ***Tifinagh*** |
| ***Vai*** |

| Central Asian Scripts |
| --- |
| ***Manichaean*** |
| ***Marchen*** |
| ***Mongolian*** |
| Mongolian Supplement |
| ***Old Sogdian*** |
| ***Old Turkic*** |
| ***Phags-Pa*** |
| ***Sogdian*** |
| ***Soyombo*** |
| ***Tibetan*** |
| ***Zanabazar Square*** |

| Indonesia & Oceania Scripts |
| --- |
| ***Balinese*** |
| ***Batak*** |
| ***Buginese*** |
| ***Buhid*** |
| ***Hanunoo*** |
| ***Javanese*** |
| ***Makasar*** |
| ***Rejang*** |
| ***Sundanese*** |
| Sundanese Supplement |
| ***Tagalog*** |
| ***Tagbanwa*** |


Emoji
Non-Emoji
120,000
80,000
40,000
0
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016

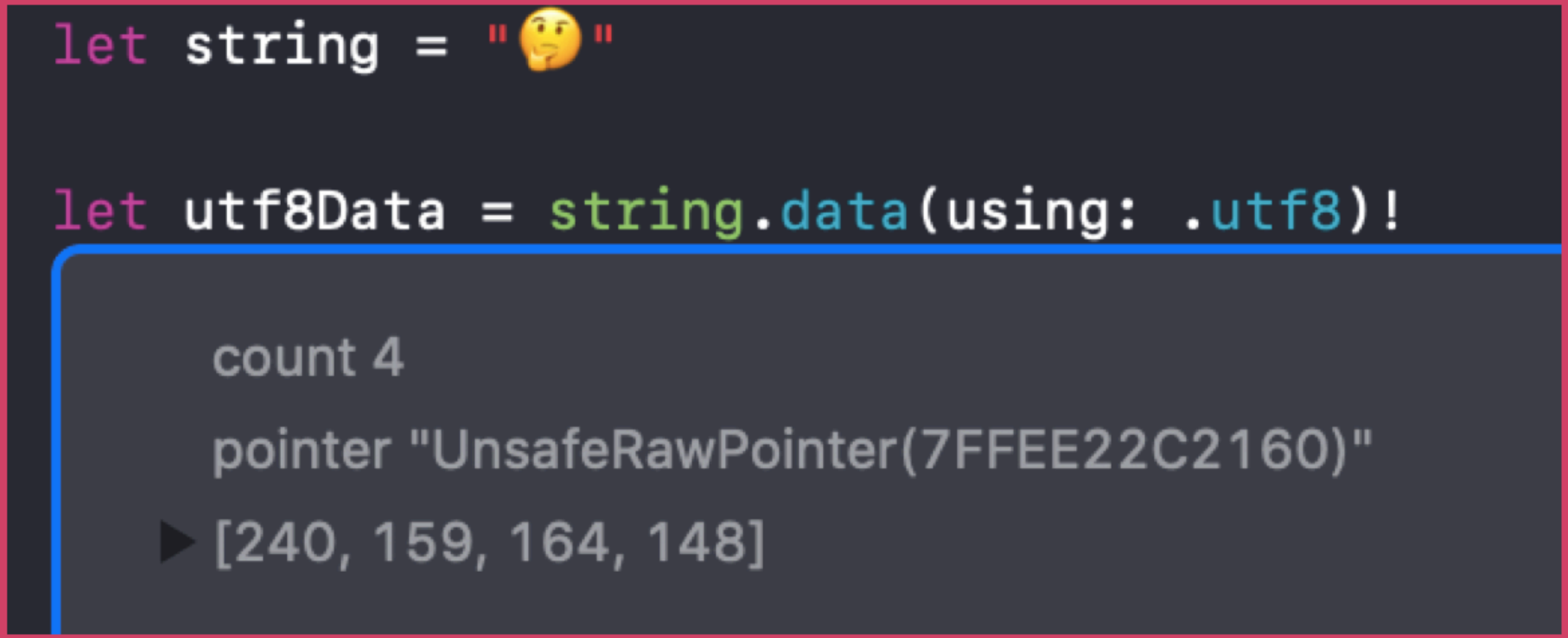
# 🤔 : U+1F914

| | |
|---|---|
| Apple |  |
| Google |  |
| Microsoft |  |
| Samsung |  |
| WhatsApp |  |
| Twitter |  |

# UTF-8

# VS.

# UTF-16


let string = "🤔"
let utf8Data = string.data(using: .utf8)!
count 4
pointer "UnsafeRawPointer(7FFEE22C2160)"
[240, 159, 164, 148]

```
let string = "🤔"

let utf16Data = string.data(using: .utf16)!
```

```
count 6
pointer "UnsafeRawPointer(7FFEE93FE180)"
▶ [255, 254, 62, 216, 20, 221]
```

"It does not make sense to have a string without knowing what encoding it uses." - Joel Spolsky

# Swift 3

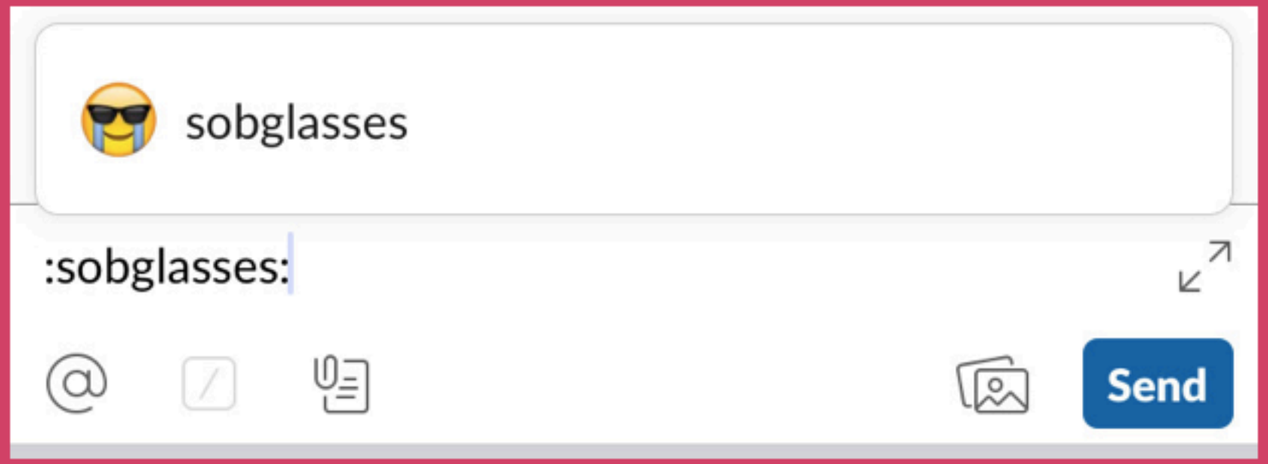
```
"🤔".characters.count = 1
"🏊".characters.count = 2
"👬".characters.count = 4
```

# Swift 4

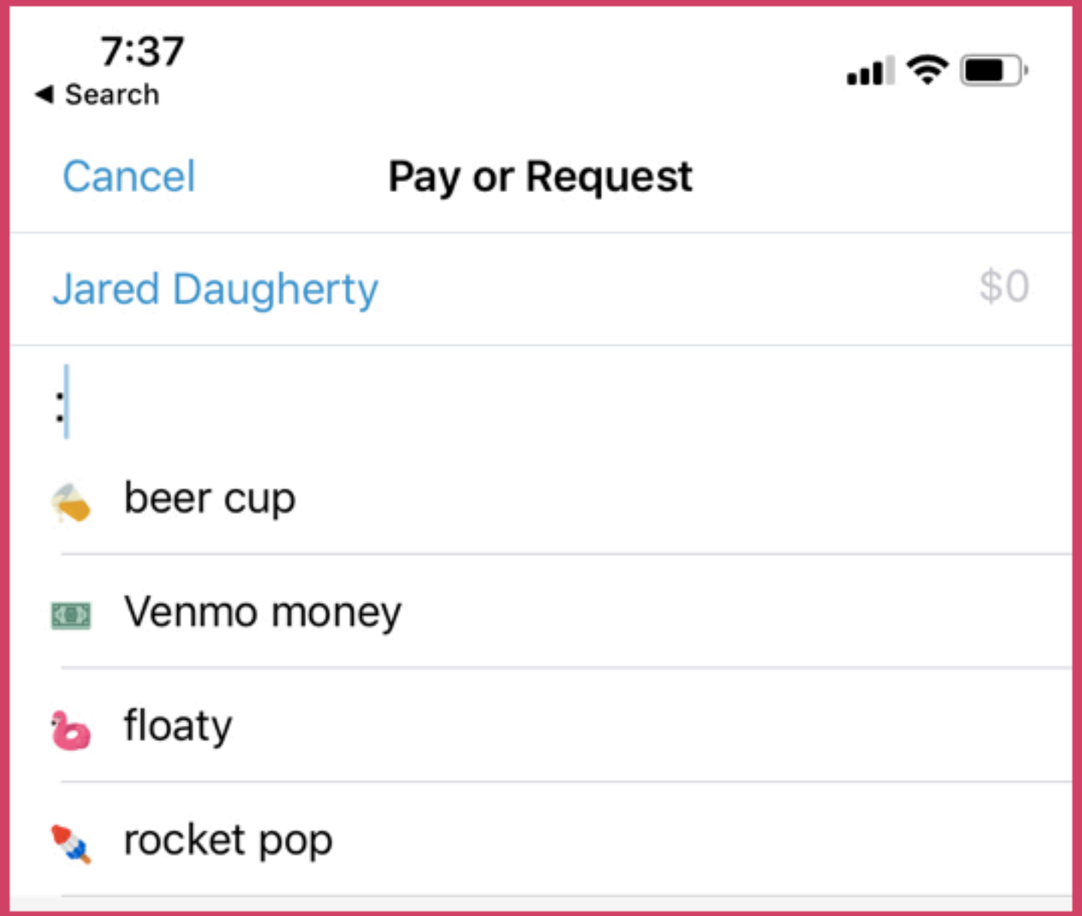
```
"🤔".count = 1
"🏊".count = 1
"👬".count = 1
```

# Custom emoji

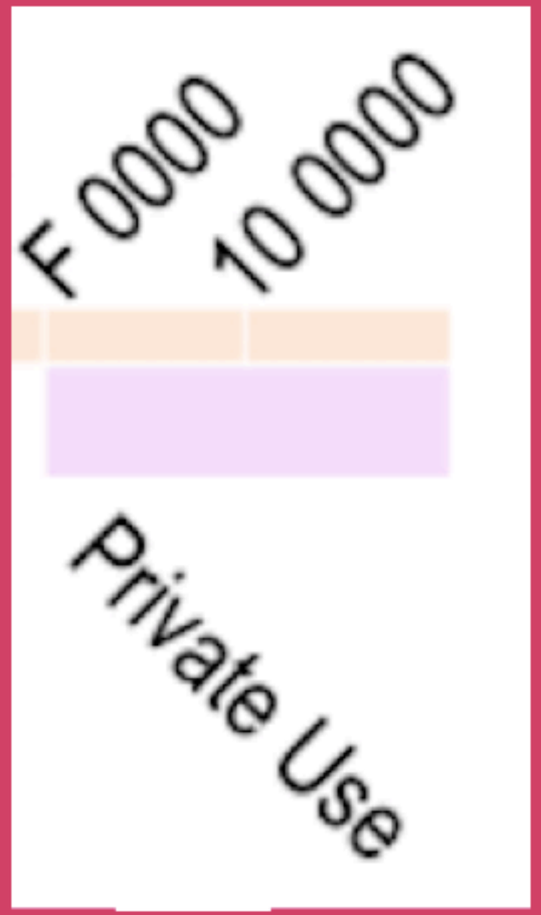
# Custom emoji




# Custom emoji



# Unicode Private Use Area



# Custom emojis in your app

1. Map human-readable string to private area codepoint
2. Generate custom font
3. Profit

# Custom emojis in your app

1. Map human-readable string to private area codepoint
2. Generate custom font
3. Profit

# 1. Map human-readable string to private area codepoint

```
{
":sobglasses:" : "\ue000"
}
```

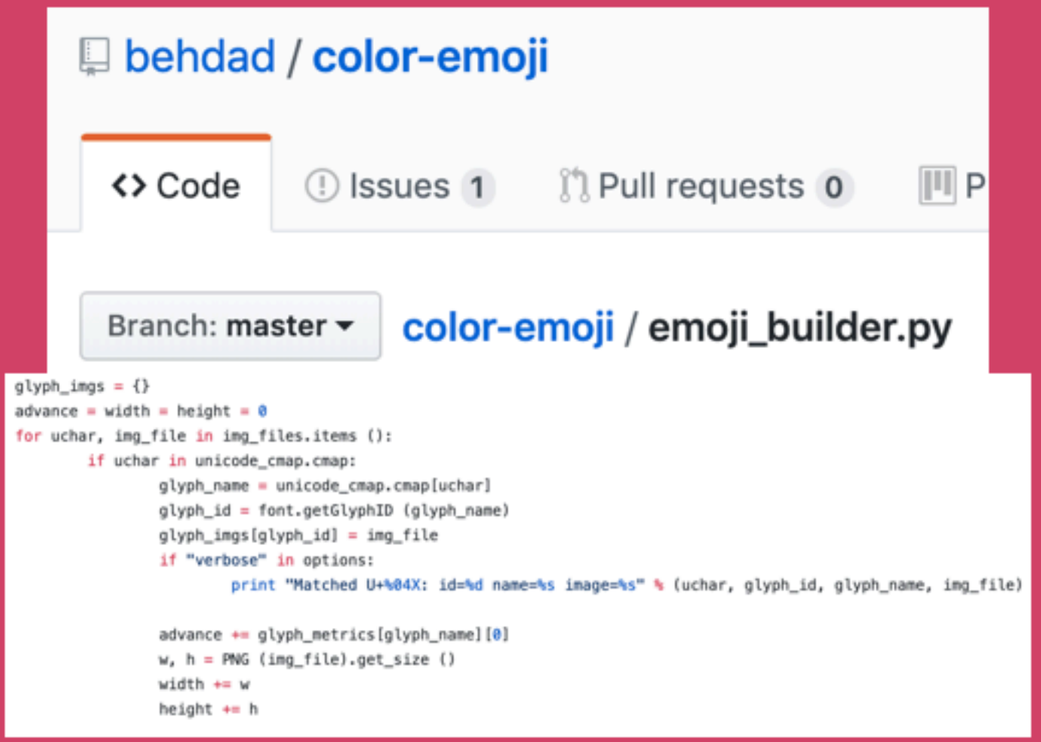
# unicode.scarfboy.com

| Some unicode data | |
|---|---|
| **Character name** | **(name not known)** |
| **Categories** | private use character, Left-to-Right |
| **Links elsewhere:** | fileformat.info/info/unicode/char/e000<br>codepoints.net/U+e000 |
| **Unicode block** | **Private Use Area**, U+E000 to U+F8FF<br>(see also the according PDF on unicode.org) |
| **Normalization** | No normalisations change the data<br>(does not necessarily mean nothing decomposes to this form) |

1. Map human-readable string to private area codepoint
2. Generate custom font
3. Profit!!!

# 2. Generate custom font

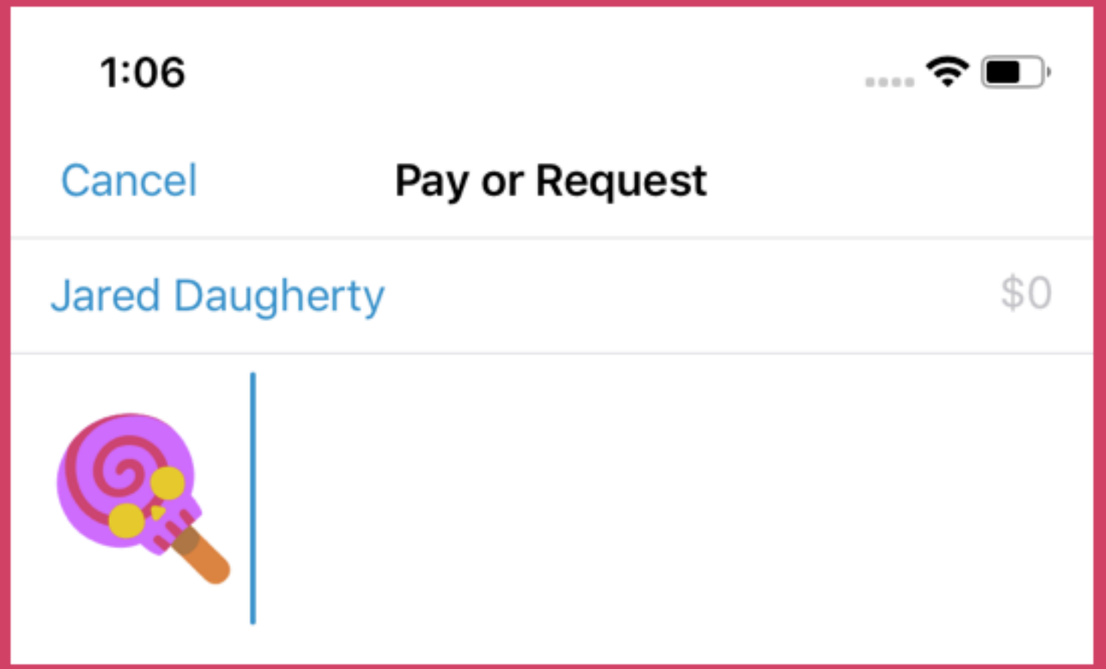
## github.com/behdad/color-emoji



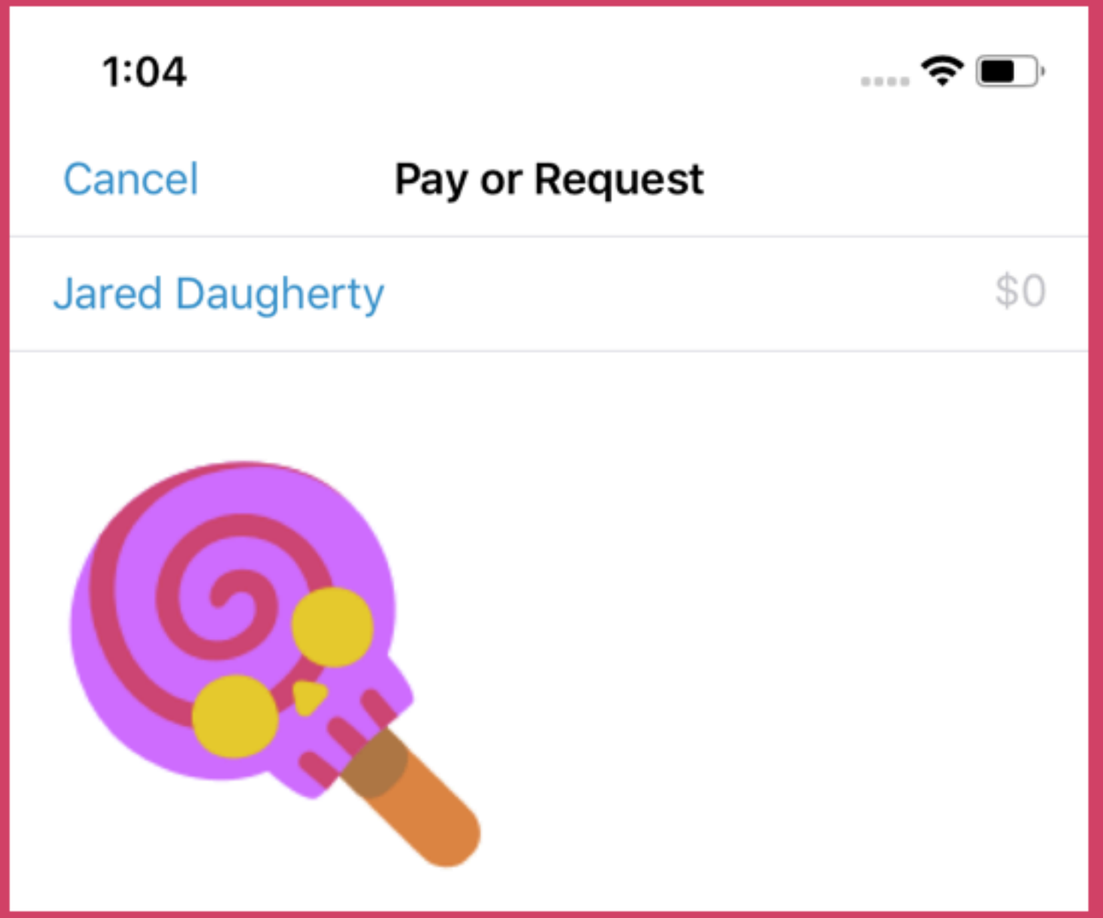
- SF Compact Text
- SF Pro Text
- Venmo Custom Emoji

# 3. Profit!!!

# Big



# BIGGER



# BIGGEST

# Summary

1. Systems of encoding
2. Unicode: code planes & code points
3. Custom emoji

# Thank you!

@shibasprinkle

# With thanks to:

**Keita Ito & Venmo team, Tim Bray, Peter Constable, Joel Spolsky & Behdad Esfahbod**