

IF YOU LIKED IT THEN YOU SHOULD HAVE PUT A STRING ON IT

360|iDev 2019

Rachel Hyman

@shibasprinkle

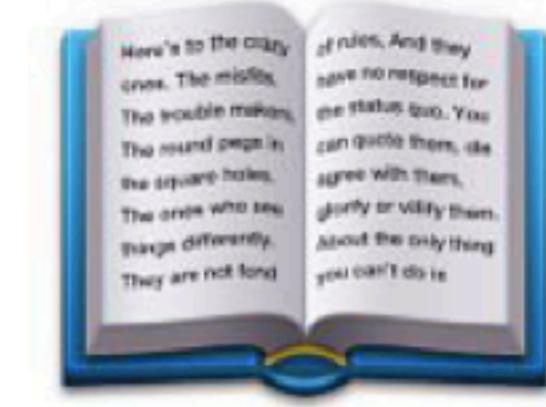
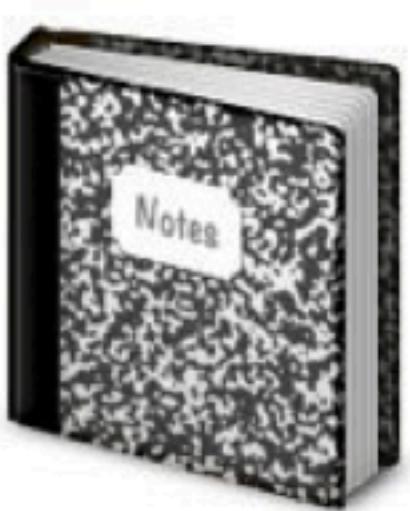
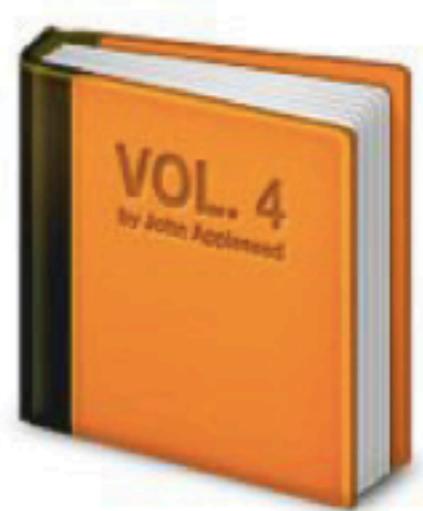
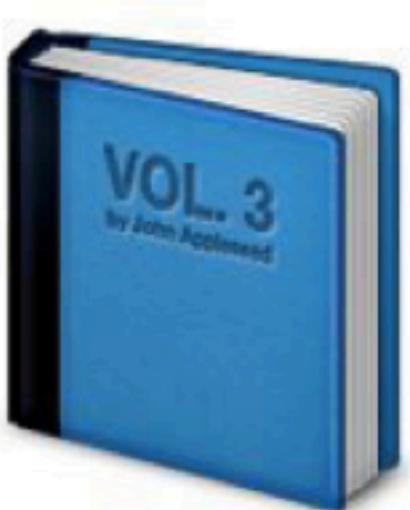
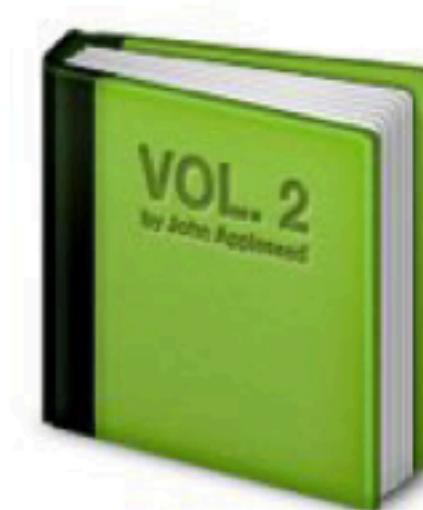
github.com/rachelhyman/emoji



venmo



Italic—Type





AN EMOJI IS
A TYPE OF STRING.

ENCODING

ENCODING: MORSE CODE

A -	J - ---	Z - -- - -
B - ····	K - ·-	T -
C - ·-·	L ·-··	U ···-
D - ···	M ---	V ····-
E - ·	N - ·---	W - ·-·-
F - ·-·-	O - ---	X - -·-·-
G - --·	P ·---·	Y - ·- - -
H ····	Q - --· -	Z - -- ···
I ···	R ···	

A

B

C

D

E

F

G

H

J

K

M

N

O

P

Q

R

S



ENCODING: ASL

ENCODING: ASCII

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	~	p
1	SOH	DC1 XON	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3 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	\		
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 I I s | o w | y t u r n e d ...

How the data might be interpreted according to another encoding:

... ז ה נ נ ת נ ו ל נ א ס ת ר ב ע נ ...

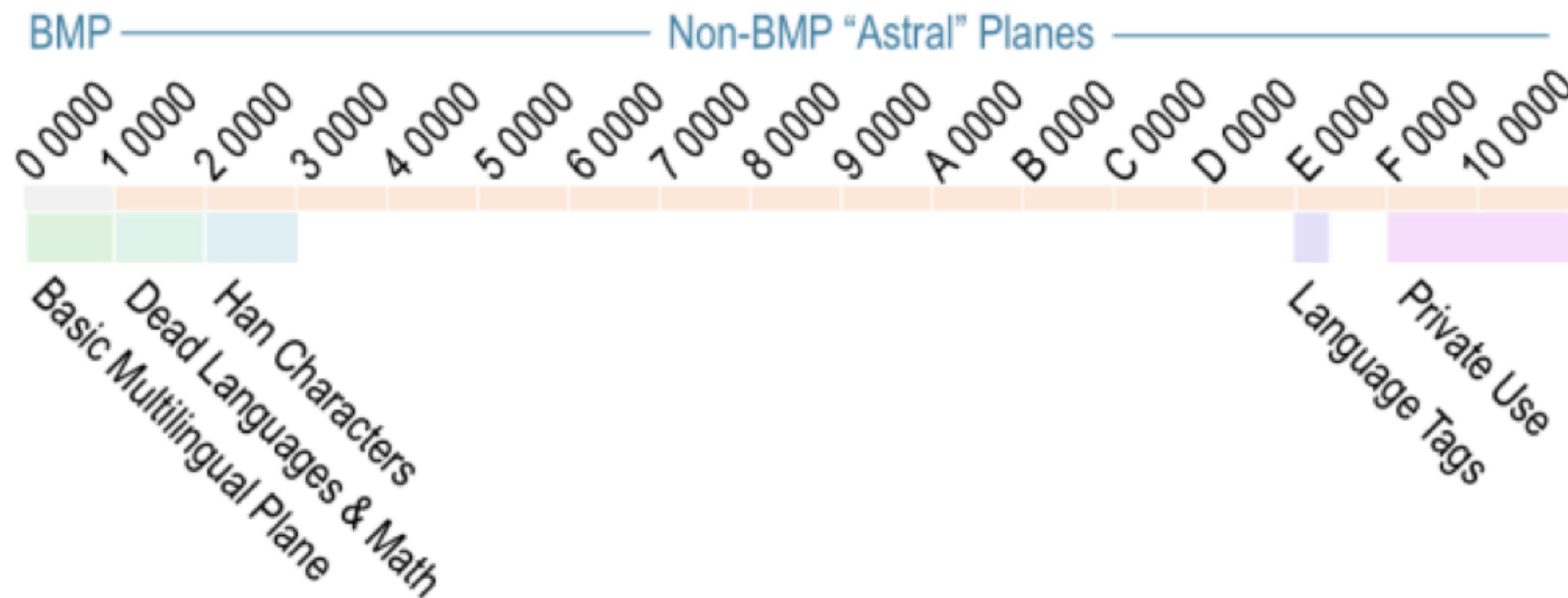
ENCODING: UNICODE

"UNICODE PROVIDES A UNIQUE NUMBER FOR EVERY CHARACTER, NO MATTER WHAT THE PLATFORM, PROGRAM, OR LANGUAGE IS."

-UNICODE

1,114,112 Unicode Code Points

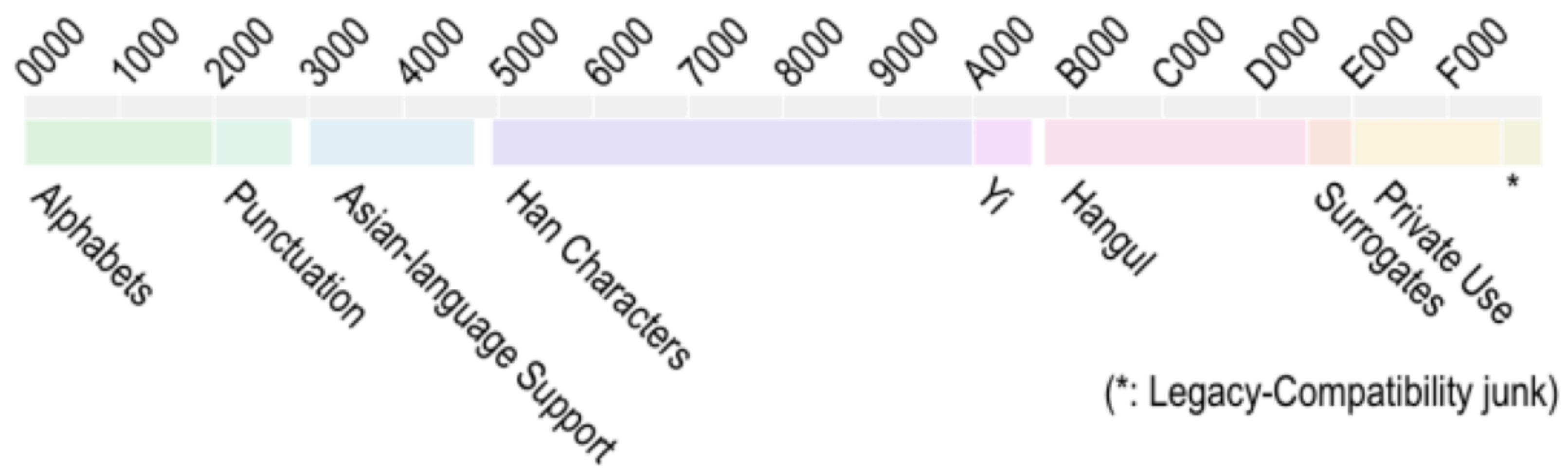
17 “Planes” each with 64k code points: U+0000 – U+10FFFF



99,024 characters defined in Unicode 5.0

The Basic Multilingual Plane (BMP)

U+0000 – U+FFFF



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

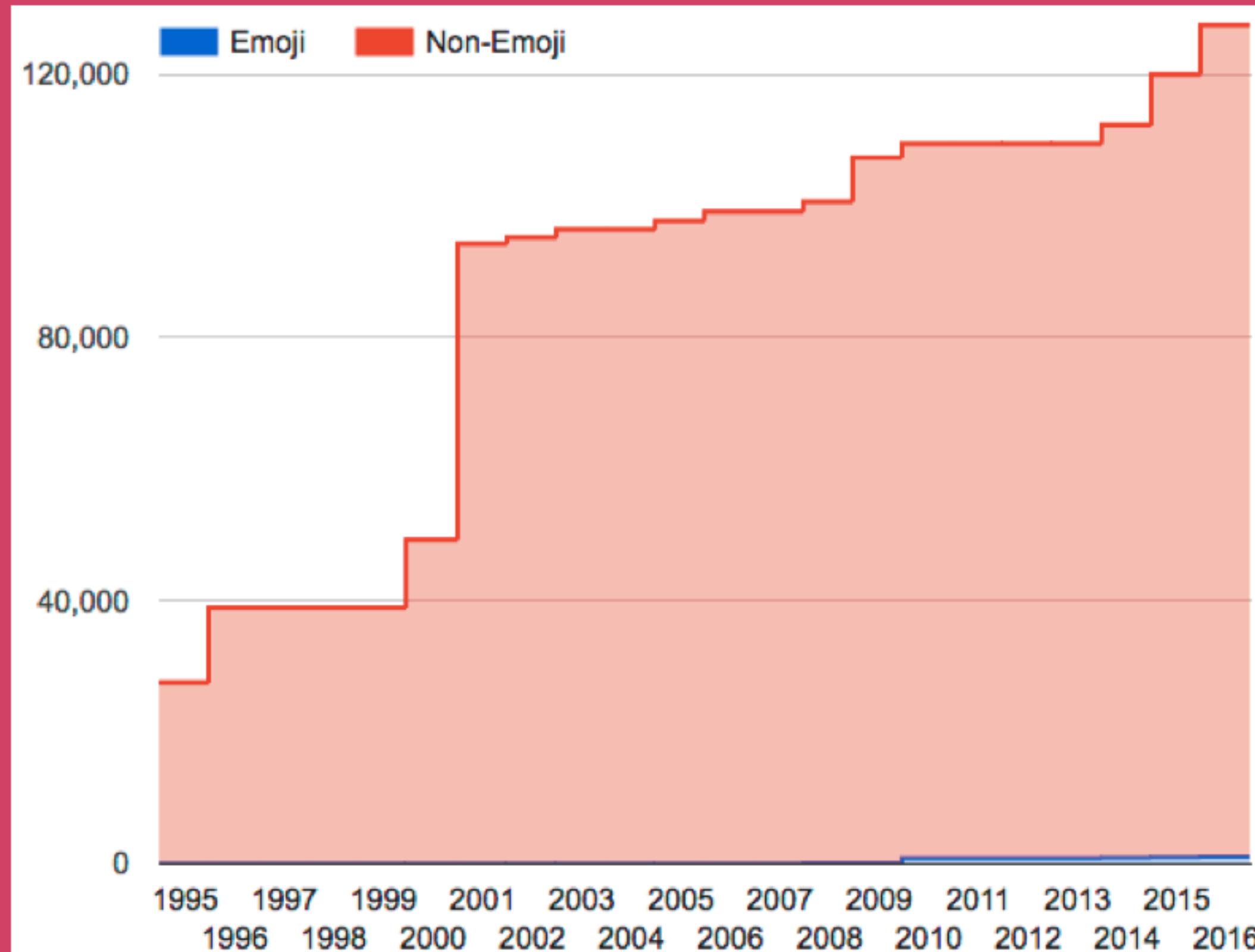
Rejang

Sundanese

Sundanese Supplement

Tagalog

Tagbanwa



🤔: 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

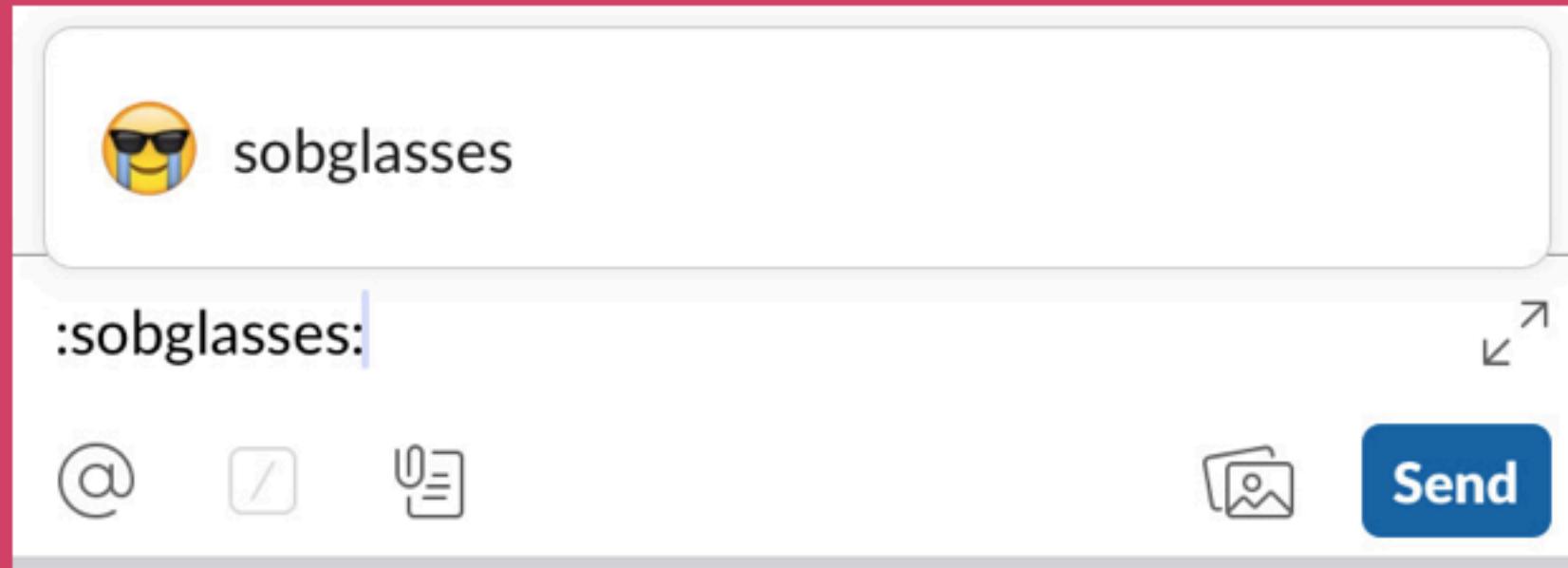
```
"🤔".characters.count = 1  
"🏊".characters.count = 2  
"👨‍👩‍👧".characters.count = 4
```

SWIFT 4

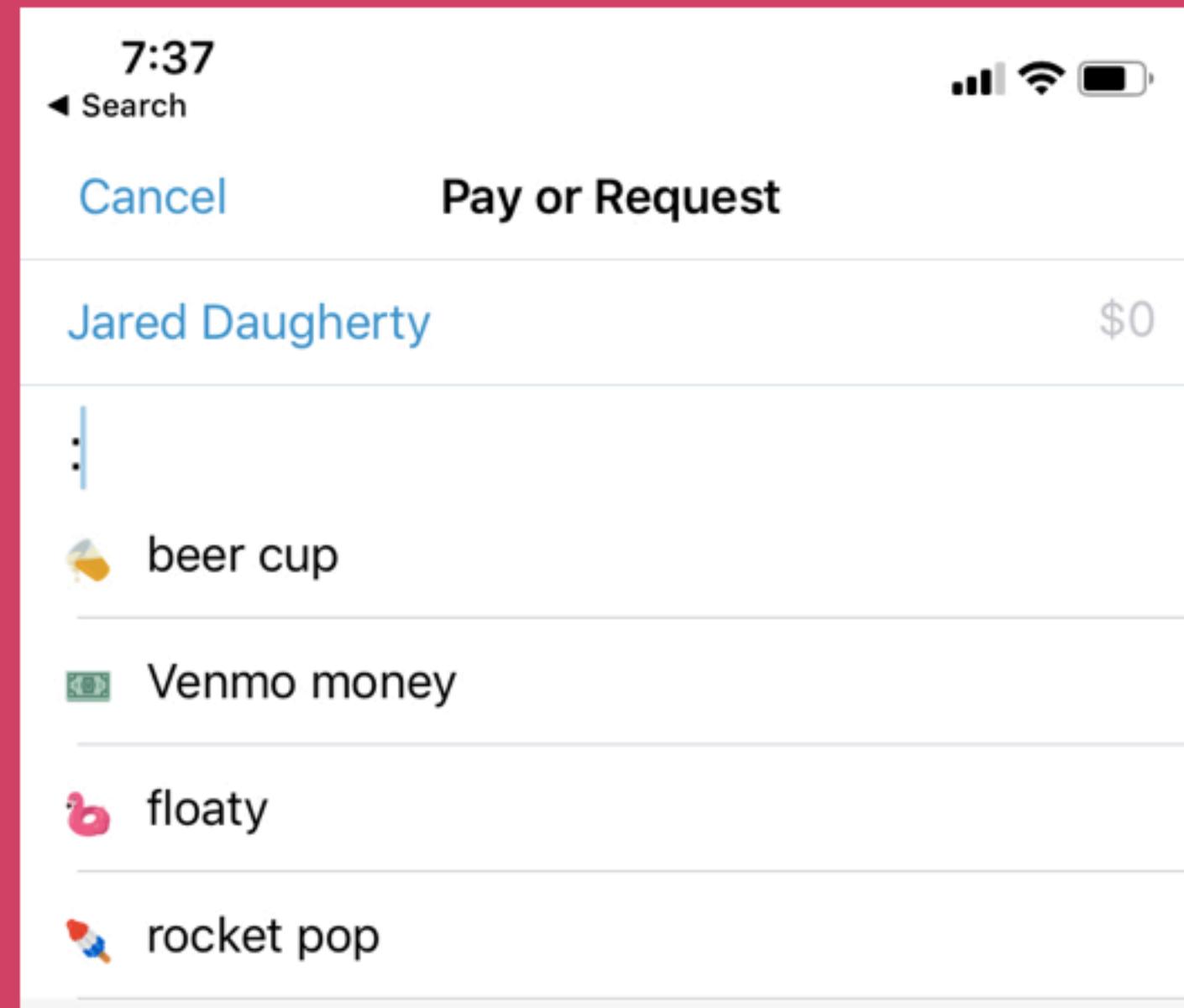
```
"🤔".count = 1  
"🏊".count = 1  
"👨‍👩‍👧".count = 1
```

CUSTOM EMOJI

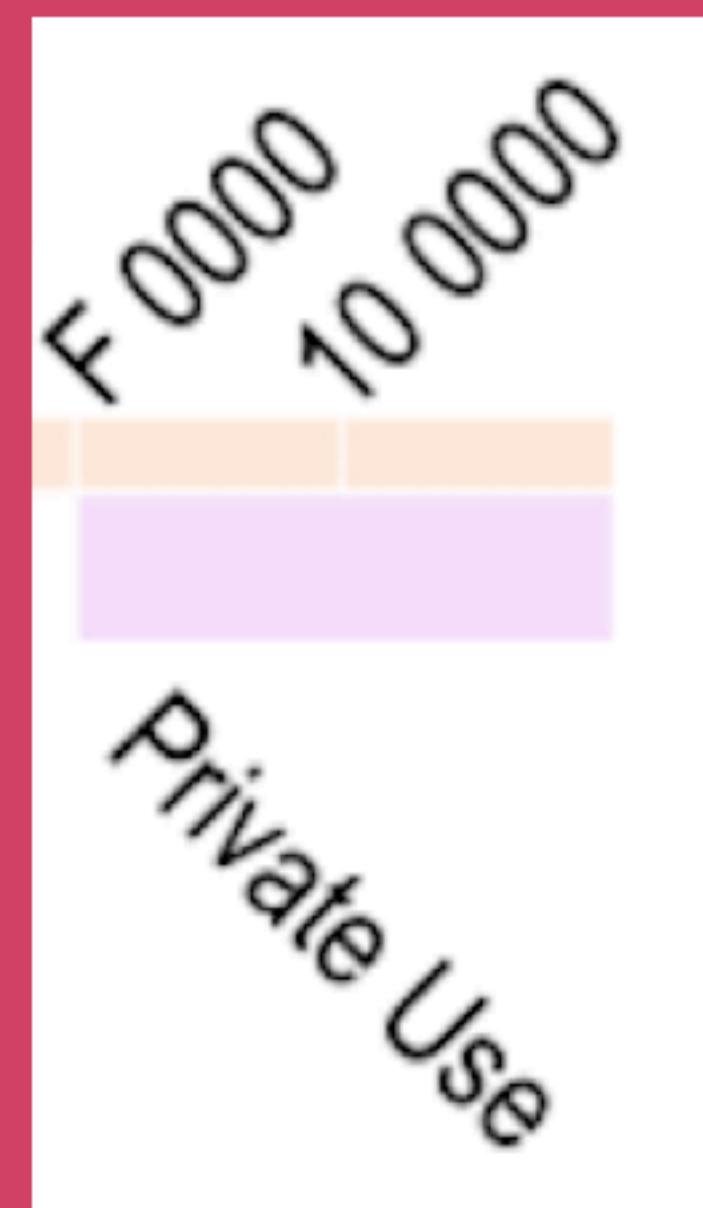
CUSTOM EMOJI



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

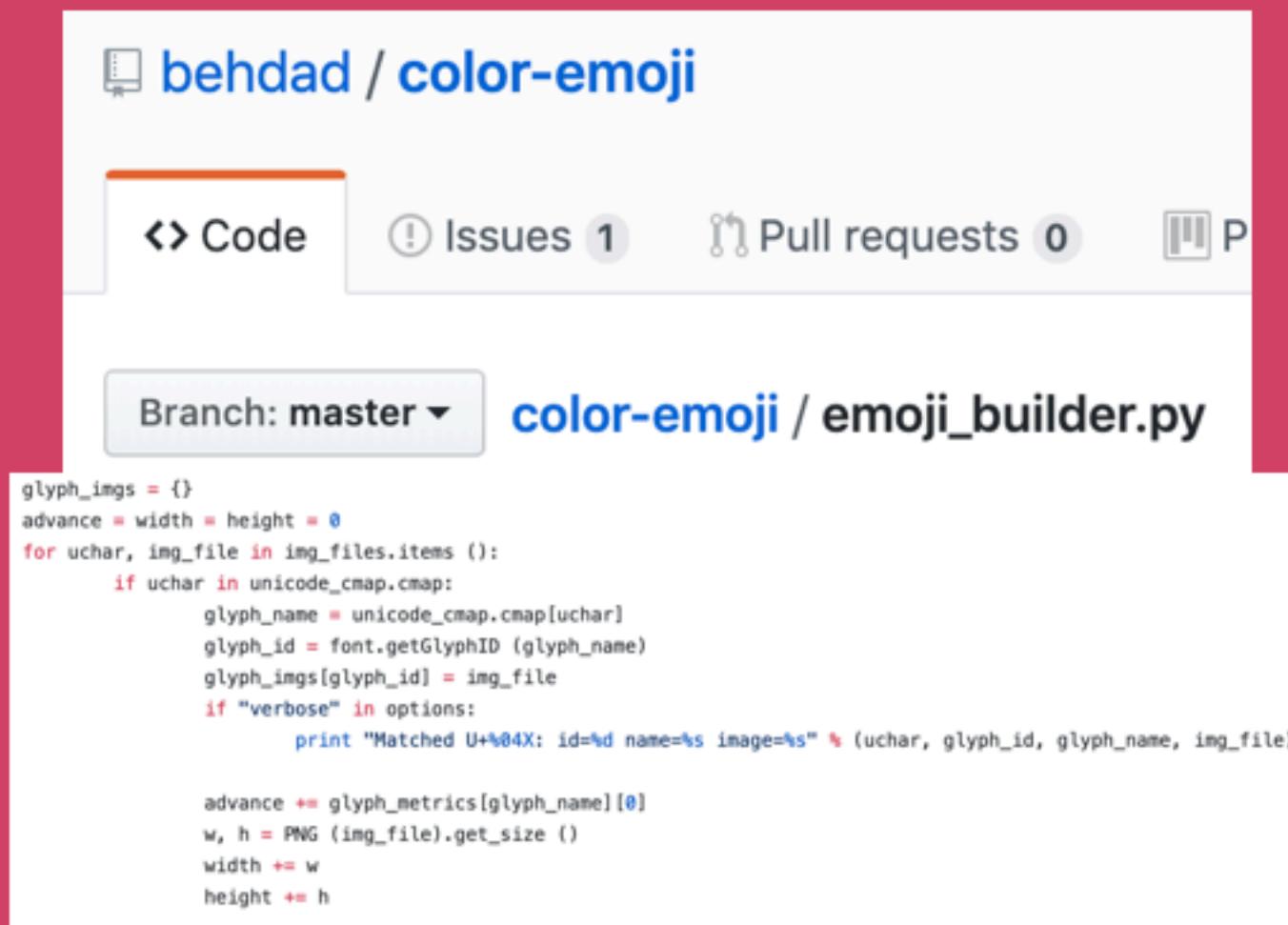
Some unicode data

Character name	(name not known)
Categories	private use character, Left-to-Right
Links elsewhere:	fileformat.info/info/unicode/char/e000 codepoints.net/U+e000
Unicode block	Private Use Area , U+E000 to U+F8FF (see also the according PDF on unicode.org)
Normalization	No normalisations change the data (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

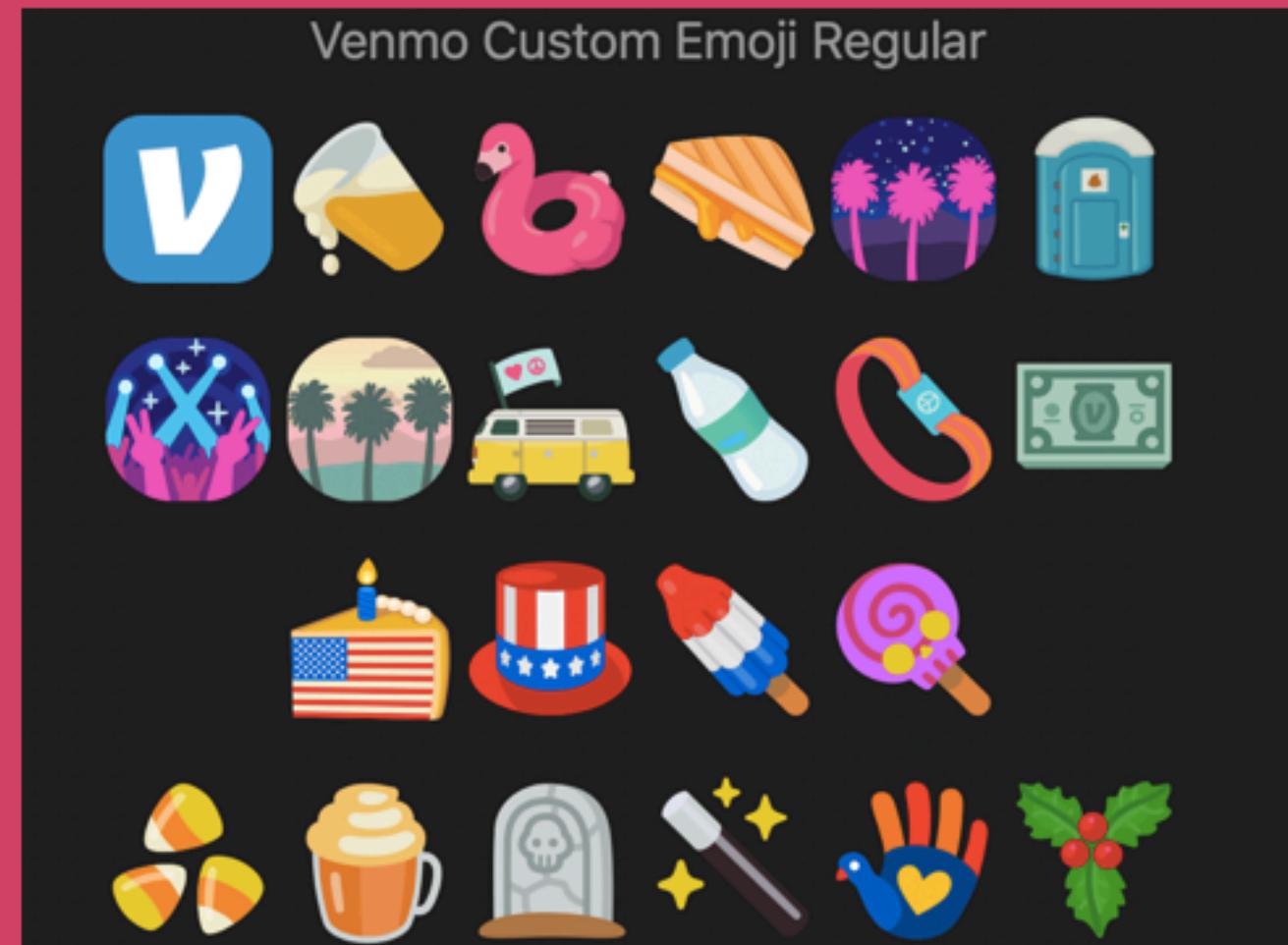
github.com/behdad/color-emoji



The screenshot shows a GitHub repository page for 'behdad / color-emoji'. The repository has 1 issue and 0 pull requests. The current branch is 'master'. The file 'color-emoji / emoji_builder.py' is displayed, containing the following Python code:

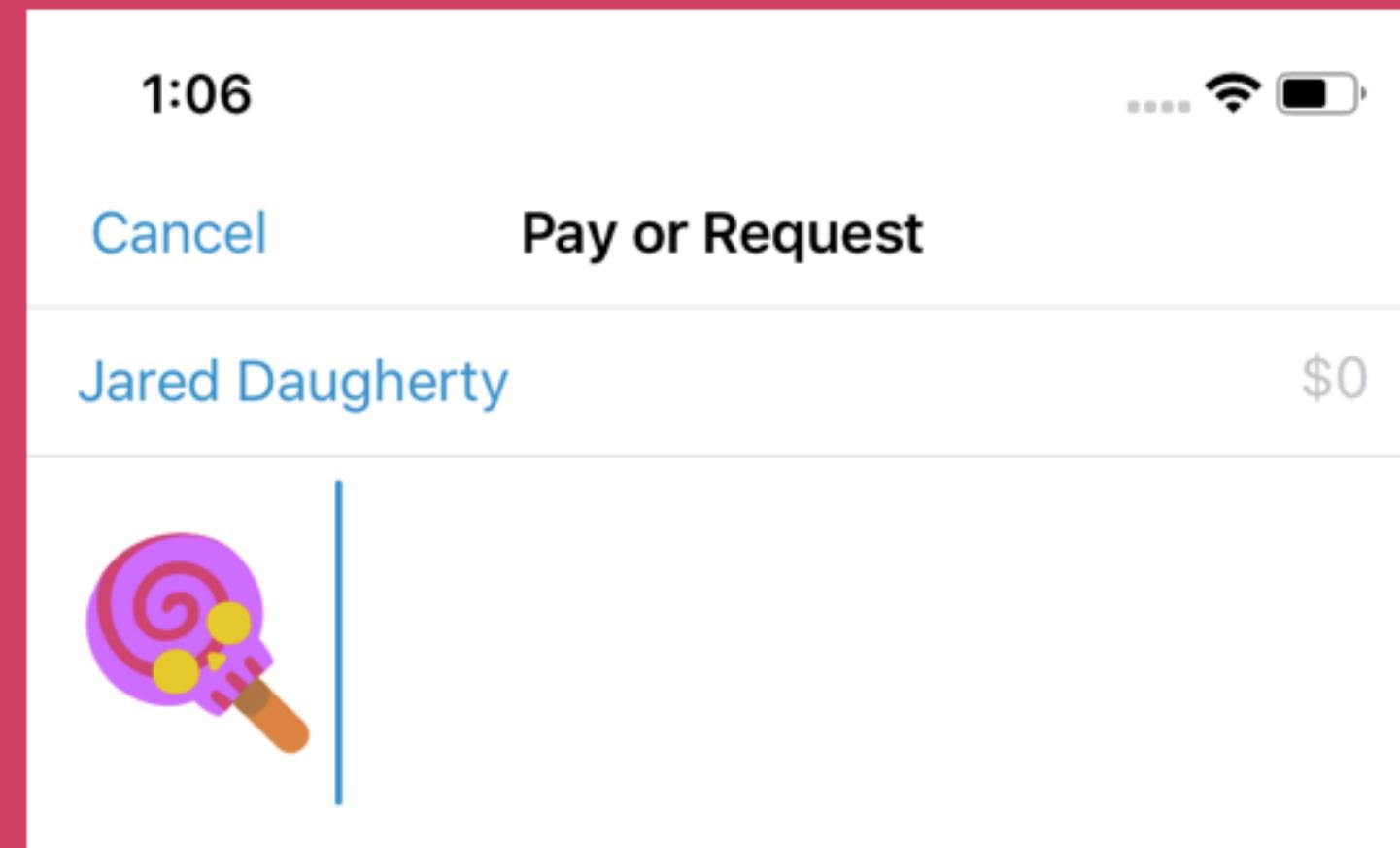
```
glyph_imgs = {}
advance = width = height = 0
for uchar, img_file in img_files.items():
    if uchar in unicode_cmap.cmap:
        glyph_name = unicode_cmap.cmap[uchar]
        glyph_id = font.getGlyphID(glyph_name)
        glyph_imgs[glyph_id] = img_file
        if "verbose" in options:
            print "Matched U+{:04X}: id={} name={} image={}".format(uchar, glyph_id, glyph_name, img_file)
        advance += glyph_metrics[glyph_name][0]
        w, h = PNG(img_file).get_size()
        width += w
        height += h
```

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

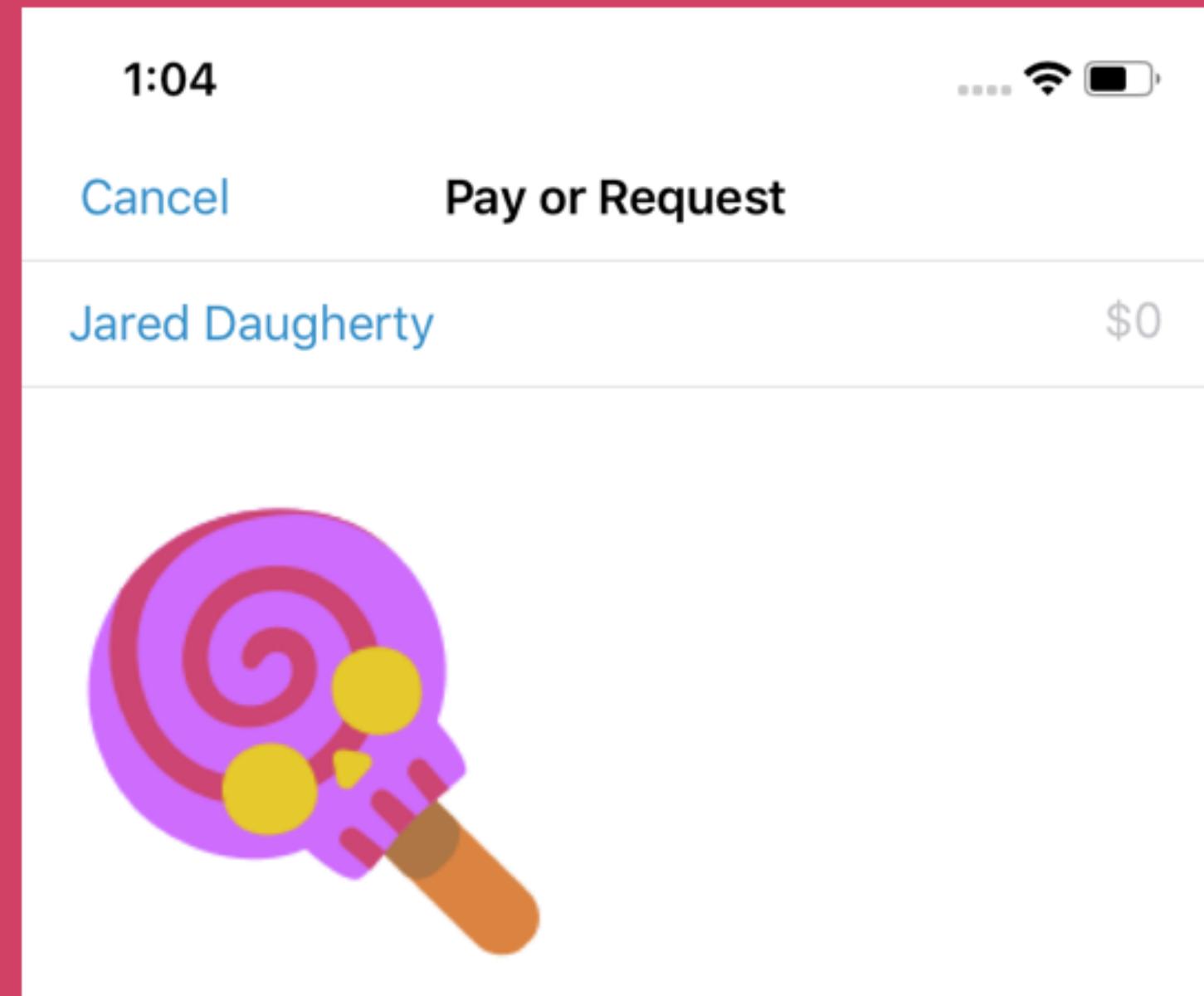


3. PROFIT!!!!

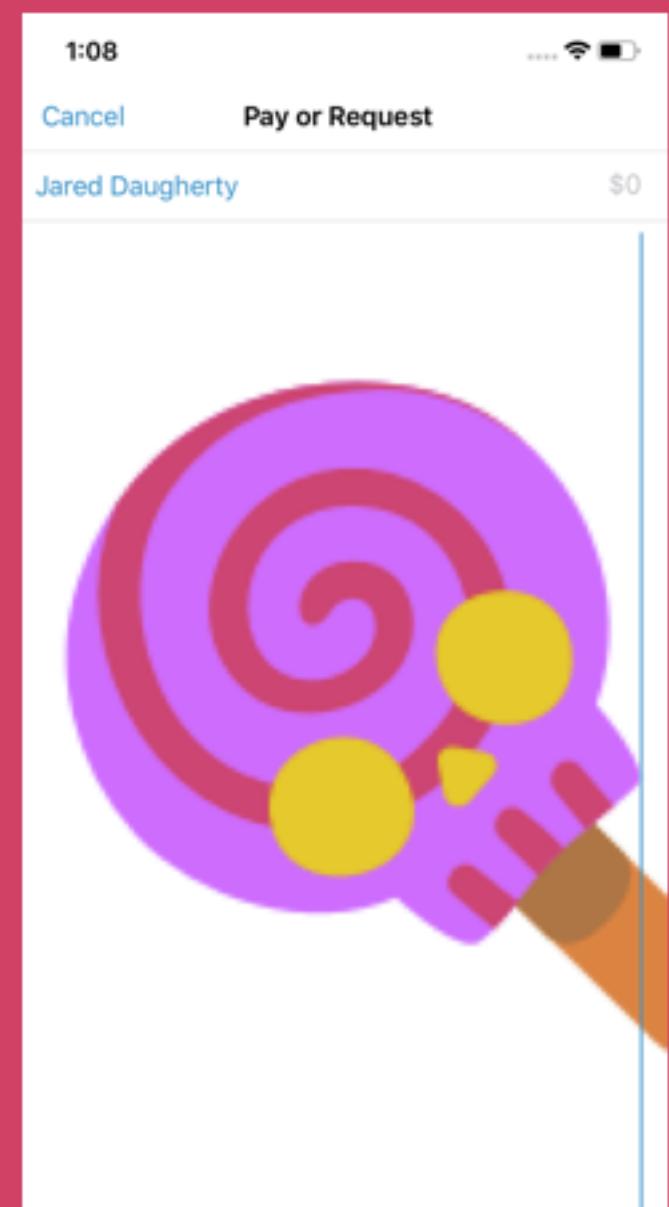
BIG



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