Android U+2764 (❤) Emoji

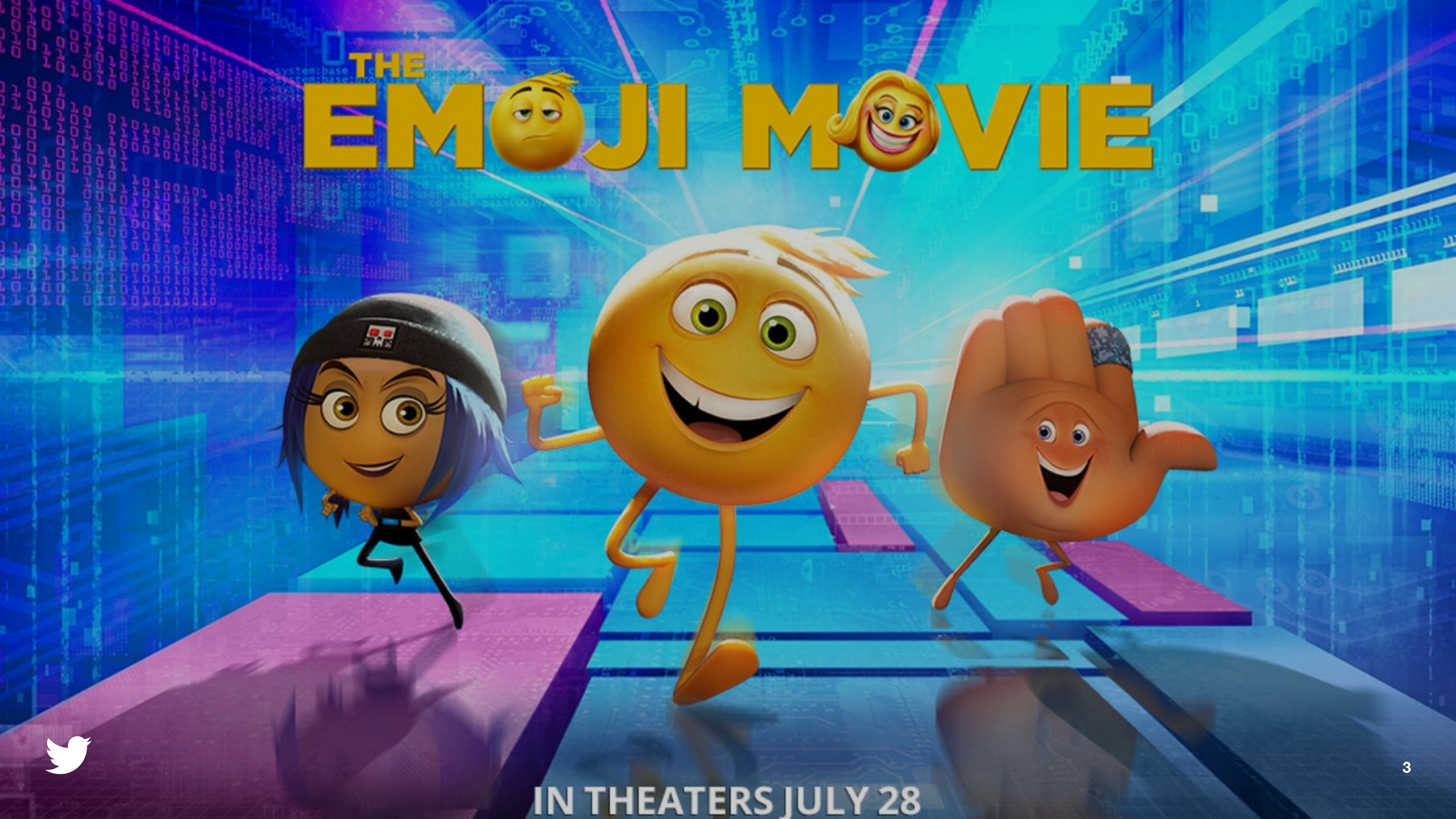
Michael Evans
@m_evans10

Oxford Dictionaries
@OxfordWords

Oxford Dictionaries Word of the Year 2015 is… 😂! http://bit.ly/20X5bA4 #OxfordWOTY

287 2K 1K

16 Nov 2015


THE
EMOJI MOVIE
IN THEATERS JULY 28

Flavor-Blasted Valerie
@valeriehalla

i wish the emoji movie was a dry documentary about the history of unicode and text encodings

30 481 1K

05 Aug 2017

History

1999

NTT DoCoMo, Shigetaka Kurita

1999

NTT DoCoMo, Shigetaka Kurita

176 characters

1999

NTT DoCoMo, Shigetaka Kurita

176 characters

“Picture character”


Free Charging
Free of Charge
ID
Password
Continuing
Clear
Search
New
Inbox:
New Acquisitions
The Original Emoji
by Shigetaka Kurita

2010

Emoji added to Unicode spec (722 characters)

2010

Emoji added to Unicode spec (722 characters)

2011: Apple added to iOS

2010

Emoji added to Unicode spec (722 characters)

2011: Apple added to iOS

2013: Google added to Android

2014

Twemoji

2014

Twemoji

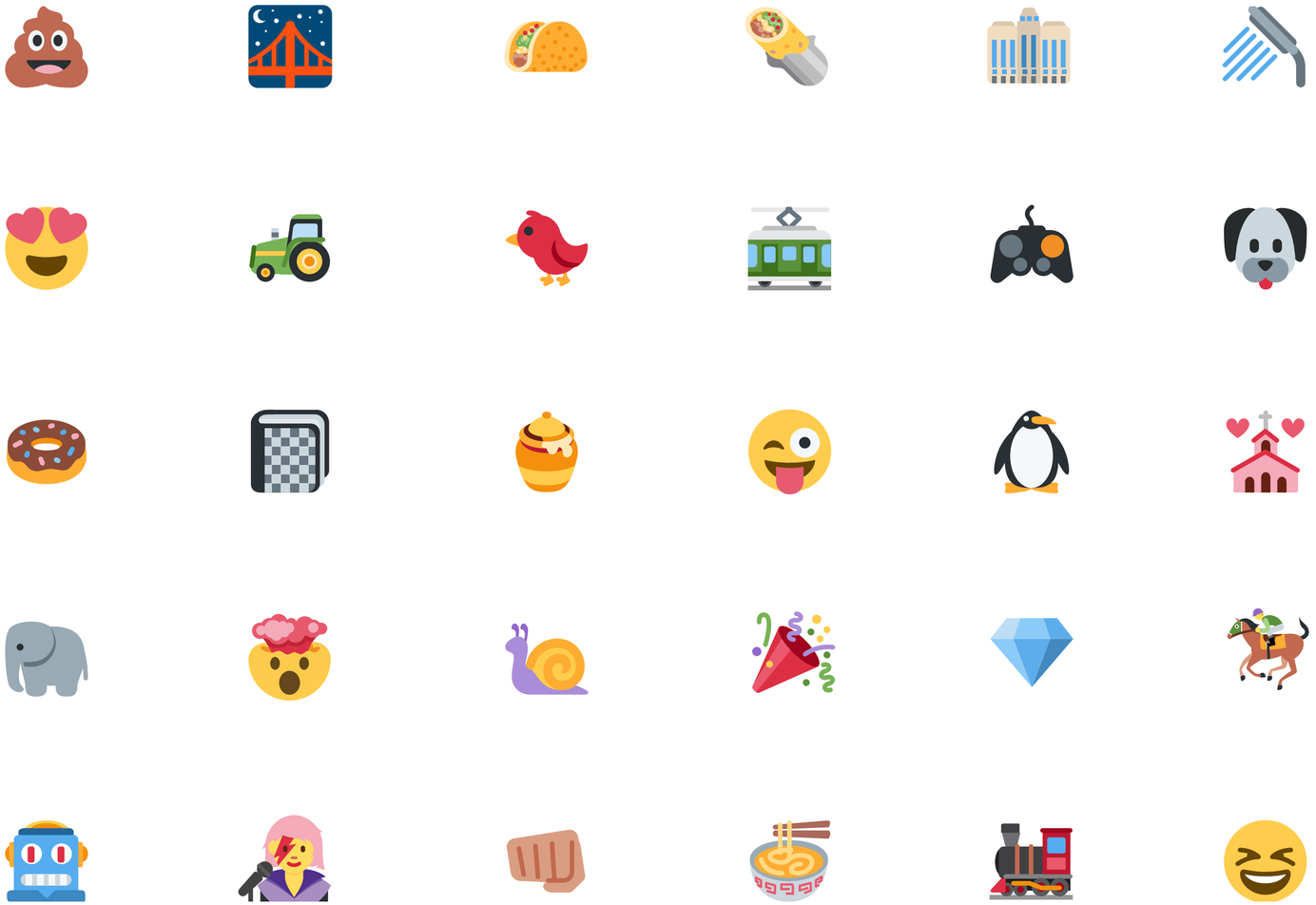
Unicode 6.1 (872 characters)

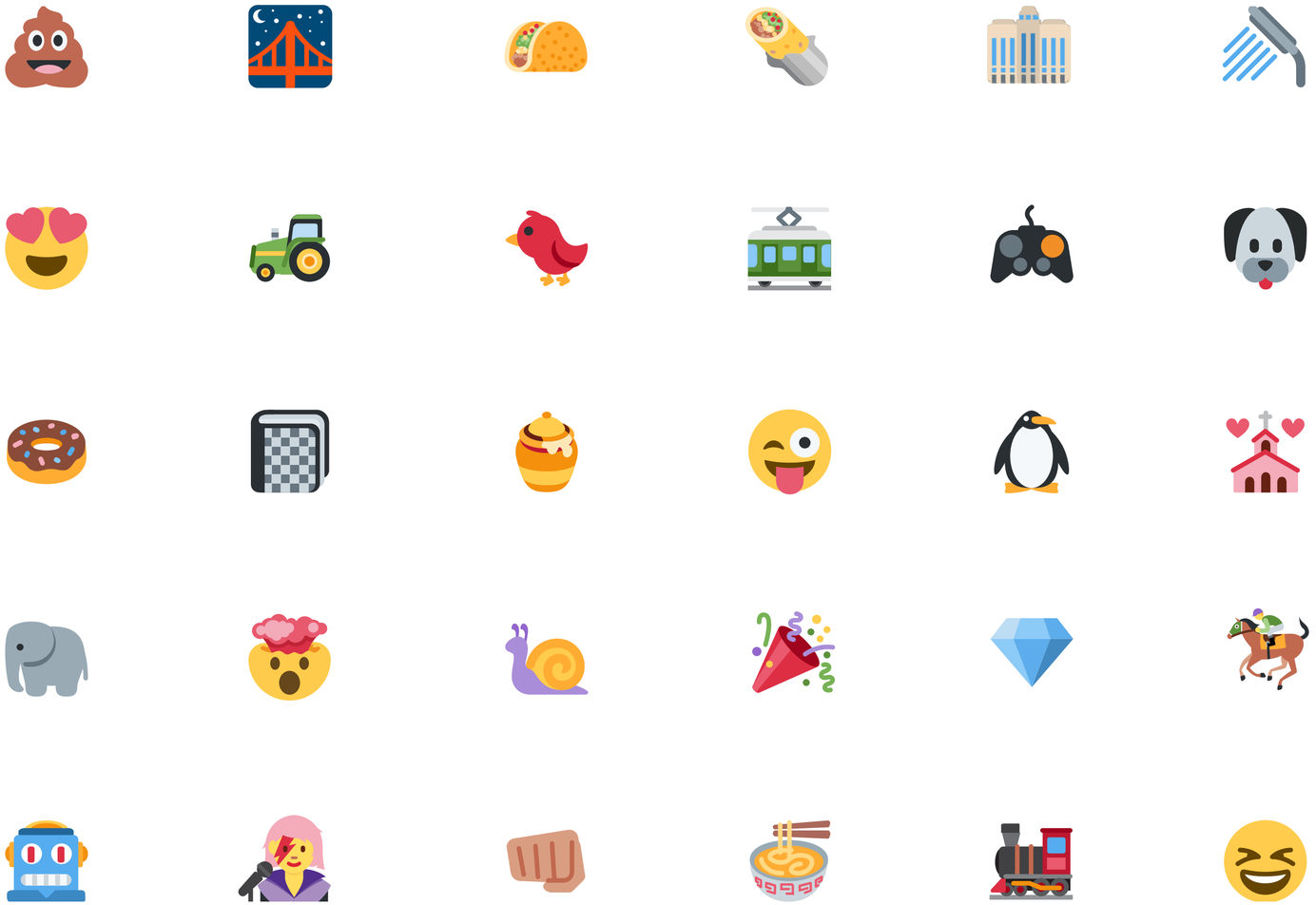
2014

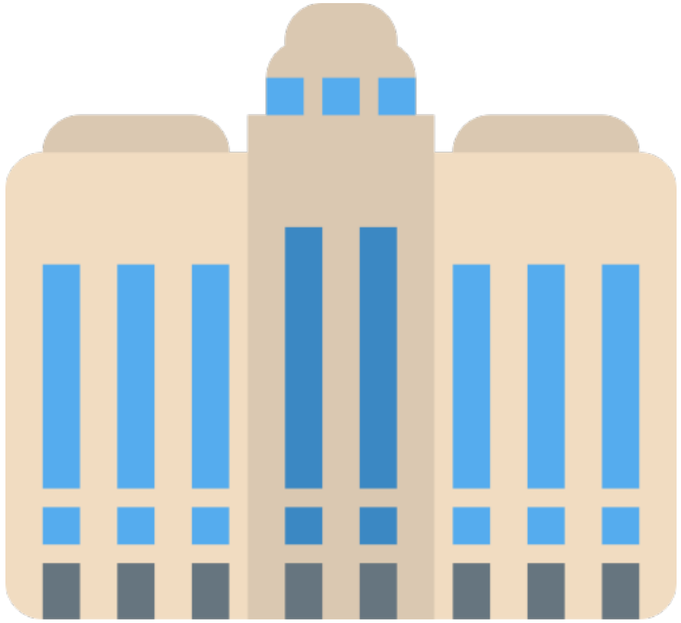
Twemoji

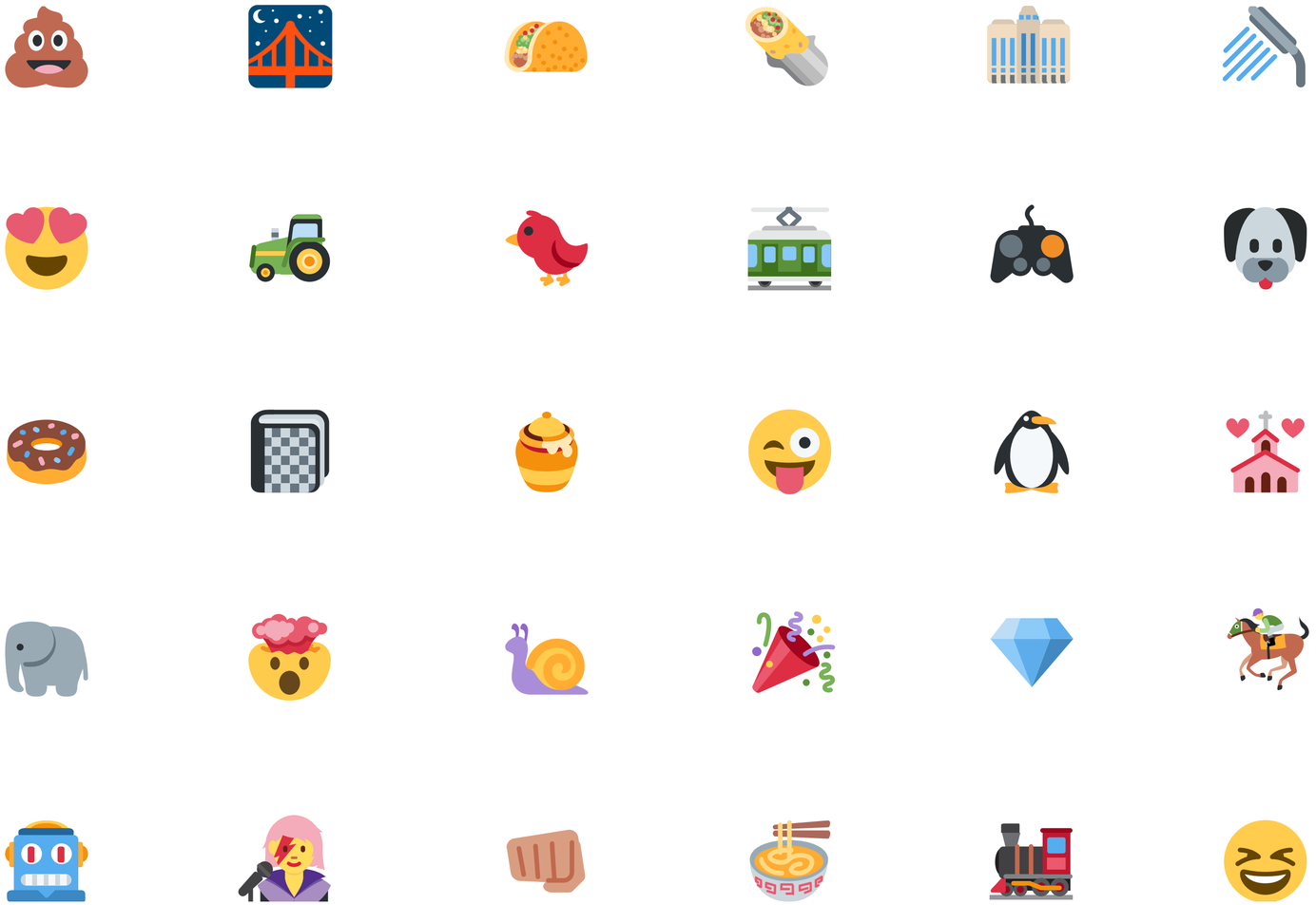
Unicode 6.1 (872 characters)

Solving □ on the web









2018

Unicode 11

2018

Unicode 11

1,644 characters

2018

Unicode 11

1,644 characters

Red / White / No hair

2018

Unicode 11

1,644 characters

Red / White / No hair
Superhero/Supervillian

2018

Unicode 11

1,644 characters

Red / White / No hair
Superhero/Supervillian
Bagel

Unicode

Unicode

Unicode

Standard for

- encoding
- representation
- handling of text

Unicode

Standard for

- encoding
- representation
- handling of text

1,112,064 code points

Java (and Kotlin)

Java (and Kotlin)

char

Java (and Kotlin)

`char`

16-bit = max value 65,535

Java (and Kotlin)

```
String example = "Hello 🌎";
for(int i = 0; i < example.length(); i++) {
  char c = example.charAt(i);
  System.out.printf("Character %d is %c%n", i, c);
}
```

Java (and Kotlin)

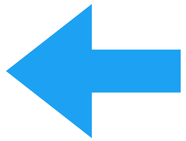
```
String example = "Hello 🌎";
for(int i = 0; i < example.length(); i++) {
  char c = example.charAt(i);
  System.out.printf("Character %d is %c%n", i, c);
}
```

```
Character 0 is H
Character 1 is e
Character 2 is l
Character 3 is l
Character 4 is o
Character 5 is
Character 6 is ?
Character 7 is ?
```

Java (and Kotlin)

```
String example = "Hello 🌎";
for(int i = 0; i < example.length(); i++) {
  char c = example.charAt(i);
  System.out.printf("Character %d is %c%n", i, c);
}
```

```
Character 0 is H
Character 1 is e
Character 2 is l
Character 3 is l
Character 4 is o
Character 5 is
Character 6 is ?
Character 7 is ?
```



Surrogate Pairs

Surrogate Pairs

Split code point across two characters

Surrogate Pairs

Split code point across two characters

Don’t use `String.charAt()`

Surrogate Pairs

```
String example = "Hello 🌎";
for(int i = 0; i < example.length(); i++) {
  char c = example.charAt(i);
  System.out.printf("Character %d is %c%n", i, c);
}
```

Surrogate Pairs

```
String example = "Hello 🌎";
for(int i = 0; i < example.length();) {
  int codePoint = example.codePointAt(i);
  System.out.printf("Character %d is %c%n", i, codePoint);
  i += Character.charCount(codePoint);
}
```

Surrogate Pairs

```
String example = "Hello 🌎";
for(int i = 0; i < example.length();) {
  int codePoint = example.codePointAt(i);
  System.out.printf("Character %d is %c%n", i, codePoint);
  i += Character.charCount(codePoint);
}
```

```
Character 0 is H
Character 1 is e
Character 2 is l
Character 3 is l
Character 4 is o
Character 5 is
Character 6 is 🌎
```

String.length()

```
>>> "😀".length
```

```
>>> "😀".length
2
```

```
>>> "😀".length
2
>>> "👍".length
```

```
>>> "😀".length
2
>>> "👍".length
2
```

```
>>> "😀".length
2
>>> "👍".length
2
>>> "👍🏻".length
```

```
>>> "😀".length
2
>>> "👍".length
2
>>> "👍🏽".length
4
```

```
>>> "😀".length
2
>>> "👍".length
2
>>> "👍🏼".length
4
>>> "👨‍👧".length
```

```
>>> "😀".length
2
>>> "👍".length
2
>>> "👍🏼".length
4
>>> "👨‍👧".length
11
```

Modifiers and Joining

Modifiers and Joining

Zero Width Joiner

Modifiers and Joining

Zero Width Joiner

U+200D

Modifiers and Joining

Modifiers and Joining

Modifiers and Joining

Modifiers and Joining

Modifiers and Joining

Modifiers and Joining

Modifiers and Joining

```
String.codePointCount()
```

Emoji on Android


I don't always understand
peralta's texts.

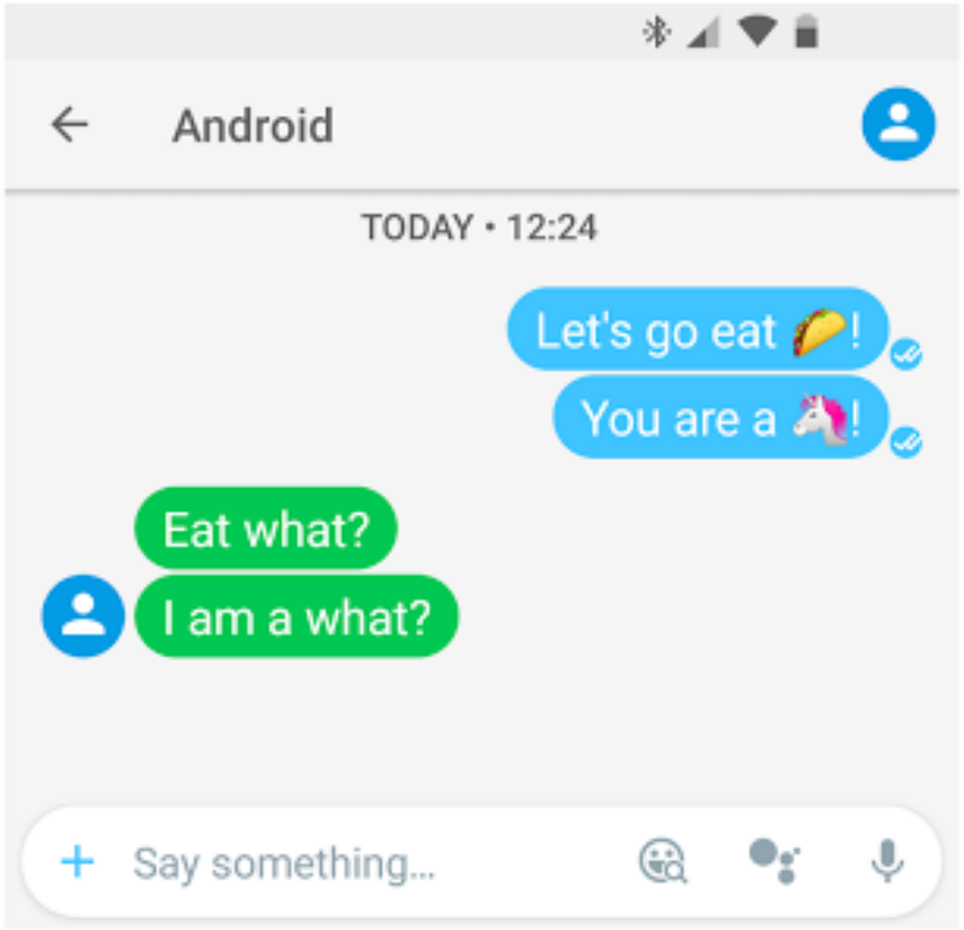
Android O

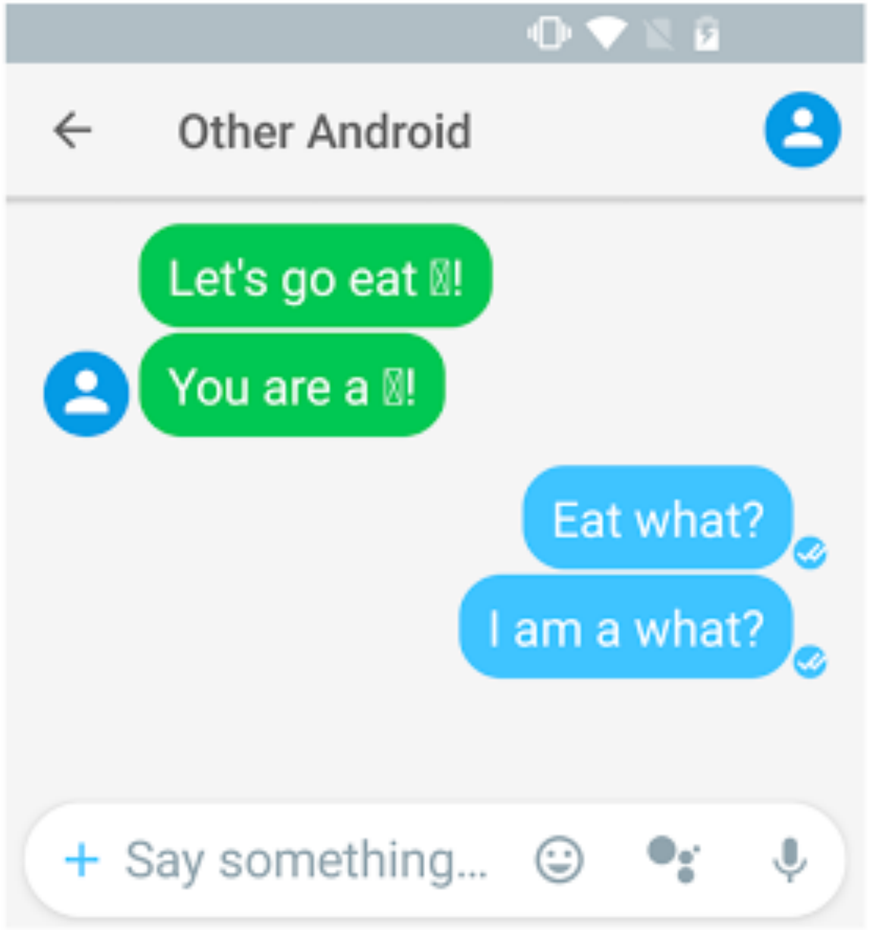
Android O

Downloadable Fonts

EmojiCompat

EmojiCompat





EmojiCompat

Bundled or downloaded

EmojiCompat

```
FontRequest fontRequest = new FontRequest(
  "com.example.fontprovider",
  "com.example",
  "emoji compat Font Query", CERTIFICATES);
  EmojiCompat.Config config = new FontRequestEmojiCompatConfig(this,
      fontRequest);
  EmojiCompat.init(config);
```

EmojiCompat

```
FontRequest fontRequest = new FontRequest(
  "com.example.fontprovider",
  "com.example",
  "emoji compat Font Query", CERTIFICATES);
  EmojiCompat.Config config = new FontRequestEmojiCompatConfig(this,
      fontRequest);
  EmojiCompat.init(config);
```

EmojiCompat

```
FontRequest fontRequest = new FontRequest(
  "com.example.fontprovider",
  "com.example",
  "emoji compat Font Query", CERTIFICATES);
  EmojiCompat.Config config = new FontRequestEmojiCompatConfig(this,
      fontRequest);
  EmojiCompat.init(config);
```

EmojiCompat

```
FontRequest fontRequest = new FontRequest(
  "com.example.fontprovider",
  "com.example",
  "emoji compat Font Query", CERTIFICATES);
  EmojiCompat.Config config = new FontRequestEmojiCompatConfig(this,
      fontRequest);
  EmojiCompat.init(config);
```

EmojiCompat

```
FontRequest fontRequest = new FontRequest(
  "com.example.fontprovider",
  "com.example",
  "emoji compat Font Query", CERTIFICATES);
  EmojiCompat.Config config = new FontRequestEmojiCompatConfig(this,
      fontRequest);
  EmojiCompat.init(config);
```

EmojiCompat

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

EmojiCompat

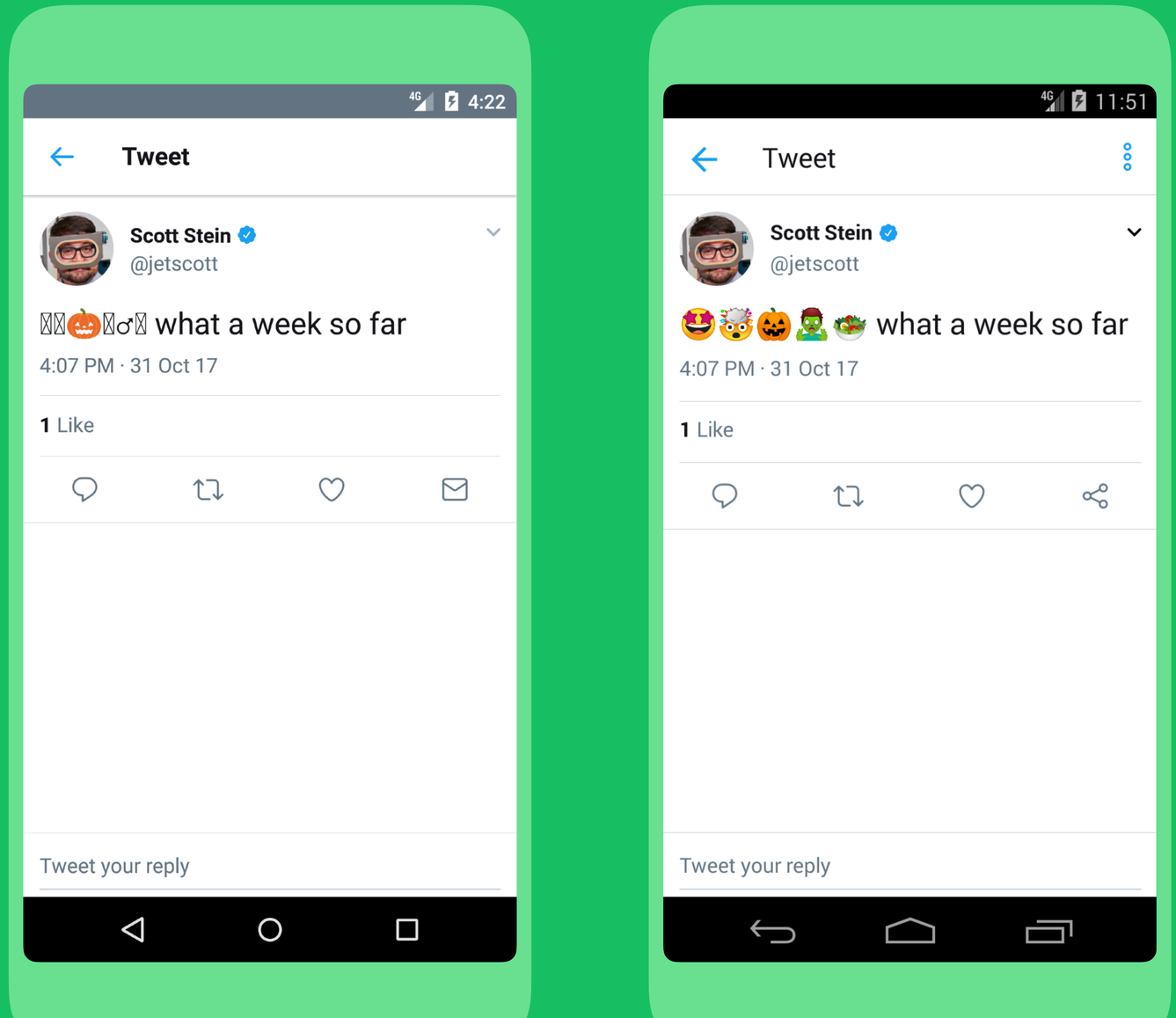
```
<android.support.text.emoji.widget.EmojiTextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

<android.support.text.emoji.widget.EmojiEditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>

<android.support.text.emoji.widget.EmojiButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

No more 🔍!

No more □!


4:22
Tweet
Scott Stein
@jetscott
□□🎃□♂□ what a week so far
4:07 PM · 31 Oct 17
1 Like
Tweet your reply
11:51
Tweet
Scott Stein
@jetscott
🤩🤯🎃🧟🥗 what a week so far
4:07 PM · 31 Oct 17
1 Like
Tweet your reply

Twemoji?

Reversing EmojiCompat

Reversing EmojiCompat

Rendering

Reversing EmojiCompat

Rendering

Configuring

Reversing EmojiCompat

Rendering

Configuring

Loading a new font

Reversing EmojiCompat

Rendering

Configuring

Loading a new font

EmojiAppCompatTextView

EmojiAppCompatTextView

```
public class EmojiAppCompatTextView extends AppCompatTextView {
    private EmojiTextViewHelper mEmojiTextViewHelper;
    private boolean mInitialized;

    private void init() {
        if (!mInitialized) {
            mInitialized = true;
            getEmojiTextViewHelper().updateTransformationMethod();
        }
    }
```

EmojiAppCompatTextView

```
public class EmojiAppCompatTextView extends AppCompatTextView {
    private EmojiTextViewHelper mEmojiTextViewHelper;
    private boolean mInitialized;

    private void init() {
        if (!mInitialized) {
            mInitialized = true;
            getEmojiTextViewHelper().updateTransformationMethod();
        }
    }
```

EmojiTextViewHelper

EmojiTextViewHelper

```
@Override
TransformationMethod wrapTransformationMethod(TransformationMethod
        transformationMethod) {
    if (transformationMethod instanceof EmojiTransformationMethod) {
        return transformationMethod;
    }
    return new EmojiTransformationMethod(transformationMethod);
}
```

EmojiTextViewHelper

```
@Override
TransformationMethod wrapTransformationMethod(TransformationMethod
        transformationMethod) {
    if (transformationMethod instanceof EmojiTransformationMethod) {
        return transformationMethod;
    }
    return new EmojiTransformationMethod(transformationMethod);
}
```

EmojiTextViewHelper

```
return EmojiCompat.get().process(source);
```

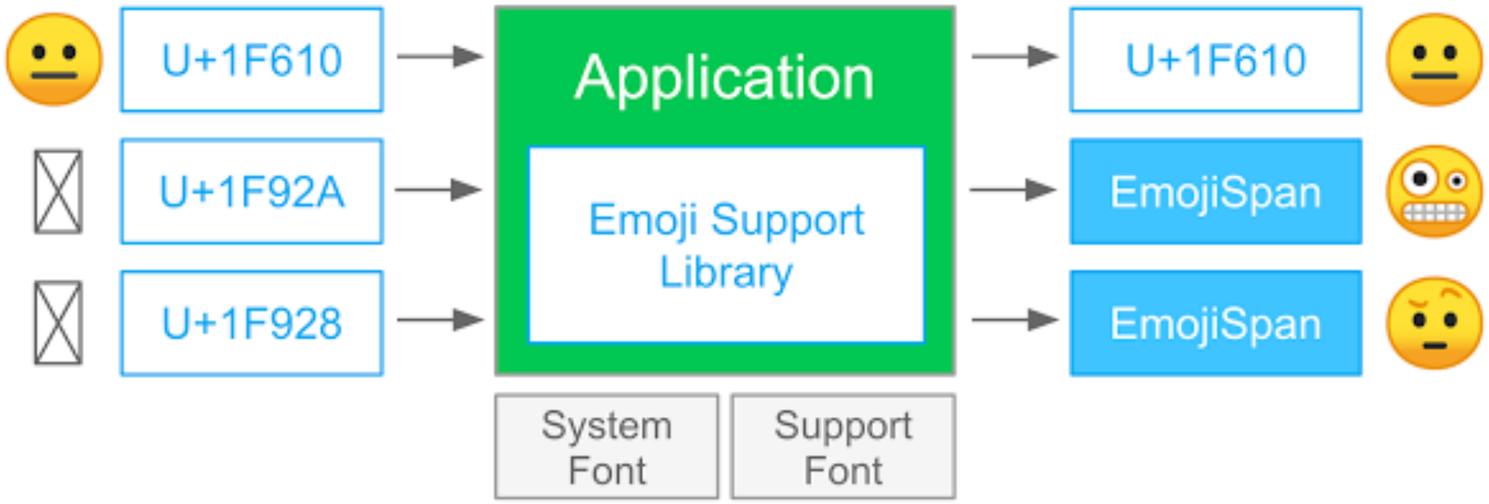
EmojiProcessor

`EmojiProcessor`

Unicode characters ➡️ `EmojiSpan`

EmojiProcessor

Unicode characters ➡️ EmojiSpan



EmojiSpan

EmojiSpan

ReplacementSpan

EmojiSpan

ReplacementSpan

Draws emoji to Canvas

Reversing EmojiCompat

Rendering

Configuring

Loading a new font

Reversing EmojiCompat

Rendering

Configuring

Loading a new font

BundledEmojiCompatConfig

BundledEmojiCompatConfig

```
try {
    final AssetManager assetManager = mContext.getAssets();
    final MetadataRepo resourceIndex = MetadataRepo.create(assetManager,
        FONT_NAME);
    mLoaderCallback.onLoaded(resourceIndex);
} catch (Throwable t) {
    mLoaderCallback.onFailed(t);
}
```

BundledEmojiCompatConfig

```
try {
    final AssetManager assetManager = mContext.getAssets();
    final MetadataRepo resourceIndex = MetadataRepo.create(assetManager,
        FONT_NAME);
    mLoaderCallback.onLoaded(resourceIndex);
} catch (Throwable t) {
    mLoaderCallback.onFailed(t);
}
```

Reversing EmojiCompat

Rendering

Configuring

Loading a new font

Reversing EmojiCompat

Rendering

Configuring

Loading a new font

Fonts

Fonts

“

Extra emoji metadata is inserted in a binary format into the font and is parsed at runtime by `EmojiCompat`. The data is embedded in the font’s `meta` table, with the private tag *Emji*.

AOSP

AOSP 🕵️

AOSP 🕵️

android / platform / external / noto-fonts / master / . / emoji-compat / **font**

```
tree: dd22872e8e5af327452b7ae1895c82485637bded [path history] [tgz]
```

NotoColorEmojiCompat.ttf

Fonts

Fonts

```
0001 0000 000d 0080 0003 0050 4342 4454
2ff6 d6c4 0000 3238 006b f7ae 4342 4c43
1fa6 ad17 006c 29e8 0000 27d4 4753 5542
b3fd 5c33 006c 51bc 0000 5f20 4f53 2f32
7604 67d7 0000 0158 0000 0060 636d 6170
89b8 f62c 0000 1ee0 0000 0de5 6865 6164
0c5c 788b 0000 00dc 0000 0036 6868 6561
1164 0ccb 0000 0114 0000 0024 686d 7478
c142 0000 0000 01b8 0000 1d26 6d61 7870
0a0f 0039 0000 0138 0000 0020 6e61 6d65
755e a57a 0000 2cc8 0000 054e 706f 7374
fb27 0084 0000 3218 0000 0020 7668 6561
0e5e 04cc 006c b0dc 0000 0024 766d 7478
```

TTX

TTX

https://github.com/fonttools/fonttools

TTX

```
➜ ttx -z extfile NotoColorEmojiCompat.ttf
```

TTX

```
➜ ttx -z extfile NotoColorEmojiCompat.ttf
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ttFont sfntVersion="\x00\x01\x00\x00" ttLibVersion="3.9">

  <GlyphOrder>
    <!-- The 'id' attribute is only for humans; it is ignored when
    parsed. -->
    <GlyphID id="0" name=".notdef"/>
    <GlyphID id="1" name="uni0000"/>
    <GlyphID id="2" name="uni000D"/>
    <GlyphID id="3" name="space"/>
    <GlyphID id="4" name="glyph00004"/>
    <GlyphID id="5" name="asterisk"/>
    <GlyphID id="6" name="glyph00006"/>
    <GlyphID id="7" name="glyph00007"/>
    <GlyphID id="8" name="glyph00008"/>
    <GlyphID id="9" name="glyph00009"/>
    <GlyphID id="10" name="glyph00010"/>
    <GlyphID id="11" name="glyph00011"/>
    <GlyphID id="12" name="glyph00012"/>
    <GlyphID id="13" name="glyph00013"/>
    <GlyphID id="14" name="glyph00014"/>
    <GlyphID id="15" name="glyph00015"/>
```

createfont.py

`createfont.py`

Adds *Emji* metadata for `EmojiCompat` support

Twemoji-Compat!


11:51
Tweet
Scott Stein
@jetscott
what a week so far
4:07 PM · 31 Oct 17
1 Like
Tweet your reply


4:45
Tweet
Scott Stein
@jetscott
what a week so far
4:07 PM · 31 Oct 17
1 Like
Tweet your reply

Available as a bundled config or a standalone font

Available as a bundled config or a standalone font

Coming *very* soon™

Questions ?

@m_evans10