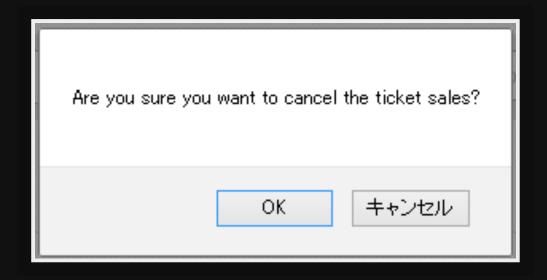
The State of the Art in Localization

@eemeli_aro github.com/eemeli





Toby took 2,566 photos on 10 July 2019

Intl.NumberFormat

```
const nf = new Intl.NumberFormat('en')
nf.format(2566)
// '2,566'
```

Intl.DateTimeFormat

```
const date = new Date(2019, 6, 10) // months are fun.
const opt = { day: 'numeric', month: 'long', year: 'numeric' }
const dtf = new Intl.DateTimeFormat('en-GB', opt)

dtf.format(date)
// '10 July 2019'
```

Intl.DateTimeFormat

```
const date = new Date(2019, 6, 10) // months are fun.
const opt = { day: 'numeric', month: 'long', year: 'numeric' }
const dtf = new Intl.DateTimeFormat('en-US', opt)

dtf.format(date)
// 'July 10, 2019'
```

Intl.RelativeTimeFormat

Chrome 71 (2018 Dec), Firefox 65 (2019 Jan), Node 12 (2019 April)

```
const rtf = new Intl.RelativeTimeFormat('en')
rtf.format(-2, 'day')
// '2 days ago'
```

Intl.ListFormat

Chrome 72 (2019 Jan)

```
const vehicles = ['Motorcycle', 'Bus', 'Car']
const opt = { style: 'long', type: 'conjunction' }
const lf = new Intl.ListFormat('en', opt)

lf.format(vehicles)
// 'Motorcycle, Bus, and Car'
```

Intl.MessageFormat?

Not yet. :(

Intl.PluralRules

```
const pr = new Intl.PluralRules('en')
pr.select(1)  // 'one' → 'photo'
pr.select(2566) // 'other' → 'photos'
```

Intl.PluralRules

```
const pr = new Intl.PluralRules('en', { type: 'ordinal' })
pr.select(1)  // 'one' → '1st'
pr.select(2)  // 'two' → '2nd'
pr.select(3)  // 'few' → '3rd'
pr.select(2566) // 'other' → '2566th'
pr.select(42)  // 'two' → '42nd'
```

Message Formatting Libraries

- FormatJS Yahoo (2014)
- I18next Locize (2012)
- messageformat OpenJS Foundation (2012)
- Project Fluent Mozilla (2017)

Front-end Environments

- React react-intl, react-i18next, @lingui/react, ...
- Angular <ng i18n>, ngx-translate, ...
- Vue vue-i18n, vue-i18next, vue-intl
- Svelte svelte-i18n

ICU MessageFormat

```
{name}'s photos

{name} took {numPhotos, plural,
    =0 {no photos}
    one {one photo}
    other {{numPhotos, number} photos}
} on {timestamp, date, long}
```

118next Translations

```
{
   "photos-title": "{{name}}'s photos",
   "photos": "one photo",
   "photos_plural": "{{count}} photos",
   "photos-taken":
    "{{name}} took $t(photos, {'count': {{numPhotos}} }) on {{tim}}
}
```

Alternatively, use i18next-icu for MessageFormat support

Polyglot.js Translations

```
{
   "photos-title": "%{name}'s photos",
   "photos": "one photo |||| %{numPhotos} photos",
   "photos-taken":
    "%{name} took %{photos_as_string} on %{timestamp_as_string}"
}
```

Project Fluent

```
photos-title = {$name}'s photos

photos-taken = {$name} took {$numPhotos ->
     [0] no photos
     [one] one photo
   *[other] {$numPhotos} photos
} on {DATETIME($timestamp,
     day: "numeric",
     month: "long",
     year: "numeric"
)}
```

Project Fluent

```
photos-title = {$name}'s photos

long-date = {DATETIME($timestamp,
    day: "numeric",
    month: "long",
    year: "numeric"
)}

photos-taken = {$name} took {$numPhotos ->
    [0] no photos
    [one] one photo
    *[other] {$numPhotos} photos
} on {long-date}
```

ICU MessageFormat

```
{name}'s photos

{name} took {numPhotos, plural,
    =0 {no photos}
    one {one photo}
    other {{numPhotos, number} photos}
} on {timestamp, date, long}
```

Messages in XLIFF

```
<xliff xmlns="urn:oasis:names:tc:xliff:document:2.0"</pre>
 version="2.0" srcLang="en-GB" trgLang="en-US">
 <file id="namespace1">
    <unit id="key1">
      <segment>
        <source>{name}'s photos
        <target>...</target>
      </segment>
   </unit>
    <unit id="key2">
      <segment>
        <source>
          {name} took {numPhotos, plural,
            =0 {no photos}
            one {one photo}
            other SsnimDhotos numbers shotoss
```

Messages in JSON

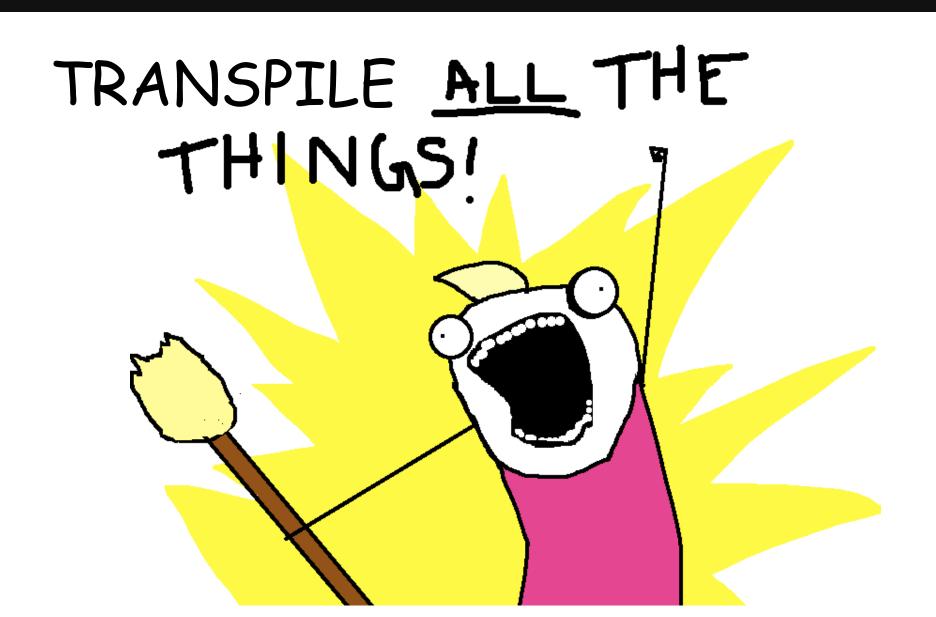
```
{
   "photos-title": "{name}'s photos",
   "photos-taken":
    "{name} took {numPhotos, plural, =0 {no photos} one {one phot}
}
```

Messages in YAML

```
photos-title: "{name}'s photos"
photos-taken: |
    {name} took {numPhotos, plural,
    =0 {no photos}
    one {one photo}
    other {{numPhotos, number} photos}
} on {timestamp, date, long}
```

Translation Services

- locize.com
- lokalise.co
- phraseapp.com
- poeditor.com
- transifex.com
- weblate.org
- ...



webpack.config.js

npm i -D messageformat messageformat-loader

Toby took 2,566 photos on 10 July 2019

```
import messages from './messages.yaml'

const msg = messages['photos-taken']
const timestamp = new Date(2019, 6, 10)

msg({ name: 'Toby', numPhotos: 2566, timestamp })
// 'Toby took 2,566 photos on 10 July 2019'
```

Toby took 2,566 photos on 10 July 2019

npm i react-message-context

```
import React from 'react'
import { Message, MessageProvider } from 'react-message-context'
import messages from './messages.yaml'
const Toby = ({ numPhotos, timestamp }) =>
 <Message id="photos-taken" name="Toby"</pre>
   numPhotos={numPhotos} timestamp={timestamp} />
export const TobyApp = () =>
 <MessageProvider messages={messages}>
   <Toby numPhotos={2566} timestamp={new Date(2019, 6, 10)} />
 </MessageProvider>
// 'Toby took 2,566 photos on 10 July 2019'
```

To start, localization doesn't have to make sense.

```
export function msg(strings, ...values) {
  if (typeof strings === 'string') return strings
  let res = ''
  for (let i = 0; i < values.length; ++i) {
    res += strings[i] + values[i]
  }
  return res + strings[strings.length - 1]
}

const name = 'Toby'
msg(name + ''s photos') // 'Toby's photos'
msg`${name}'s photos` // 'Toby's photos'</pre>
```

It can be just as dumb in React.

```
import React from 'react'

export const Msg = ({ children }) => children

function TobyTitle() {
  const name = 'Toby'
  return <Msg>{name}'s photos</Msg>
}
```

Kotoistamisen huippuluokka

@eemeli_aro github.com/eemeli

