TC 39

JS Intl API 2018

Lists, Relative Time, Units, Segmenting and more...

9.12.2018

Zibi Braniecki
Platform Engineer at Mozilla

Agenda

1. ECMA-402 5th edition

   Summary of the ECMAScript 2018 Internationalization API Specification

2. Intl.Locale

   en-Latn-CA-u-ca-buddhist

3. Intl.ListFormat

   Anna, John and Mary

4. Intl.NumberFormat 2

   9.81 m/s²

5. Intl.RelativeTimeFormat

   21 minutes ago

6. … and more

ECMA-402 5th Edition

Intl.PluralRules

Intl.NumberFormat.formatToParts

hourCycle / -u-hc-*

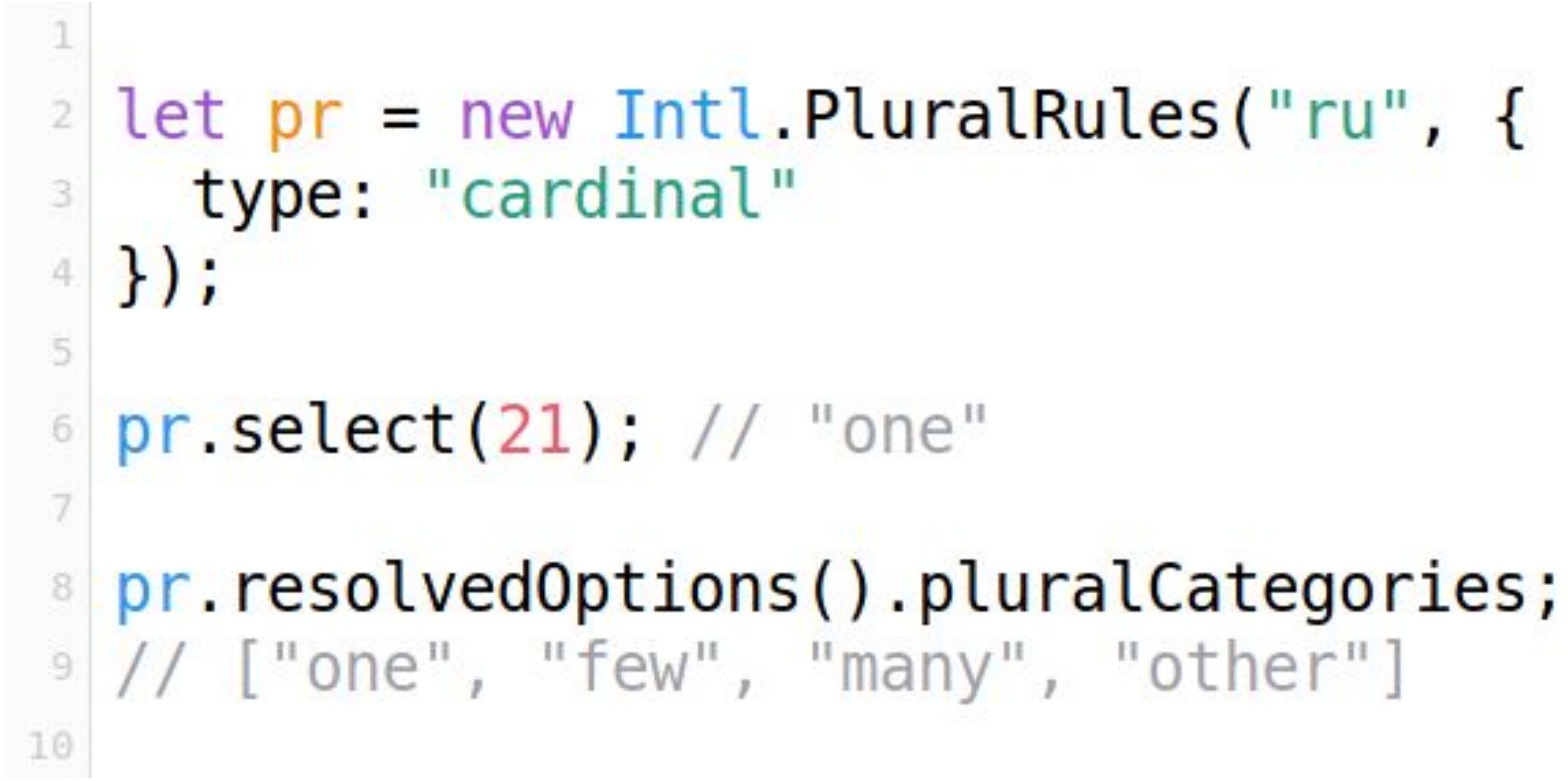
Intl.PluralRules

New API

- Low level pluralization support
- Requirement for many custom Intl.Formatters
- in particular, requirement for any l10n API

Intl.PluralRules

New API



```
let pr = new Intl.PluralRules("ru", {
  type: "cardinal"
});

pr.select(21); // "one"

pr.resolvedOptions().pluralCategories;
// ["one", "few", "many", "other"]
```

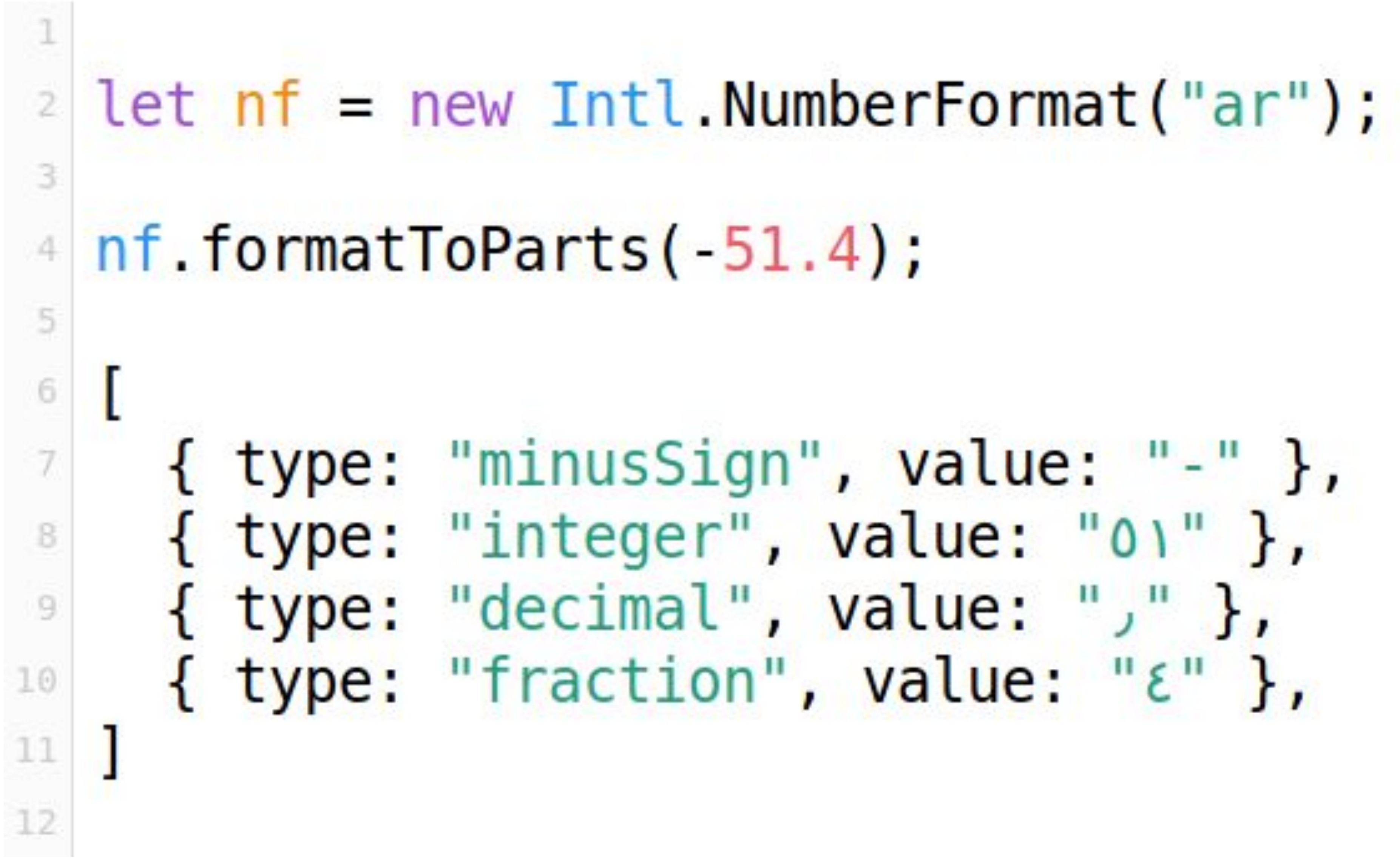
Intl.NumberFormat.formatToParts

New API

- “mid”-level data operations
- intl-transparent styling
- intl-transparent customizations

Intl.NumberFormat.formatToParts

New API



```
let nf = new Intl.NumberFormat("ar");

nf.formatToParts(-51.4);

[
  { type: "minusSign", value: "-" },
  { type: "integer", value: "٥١" },
  { type: "decimal", value: "٫" },
  { type: "fraction", value: "٤" },
]
```

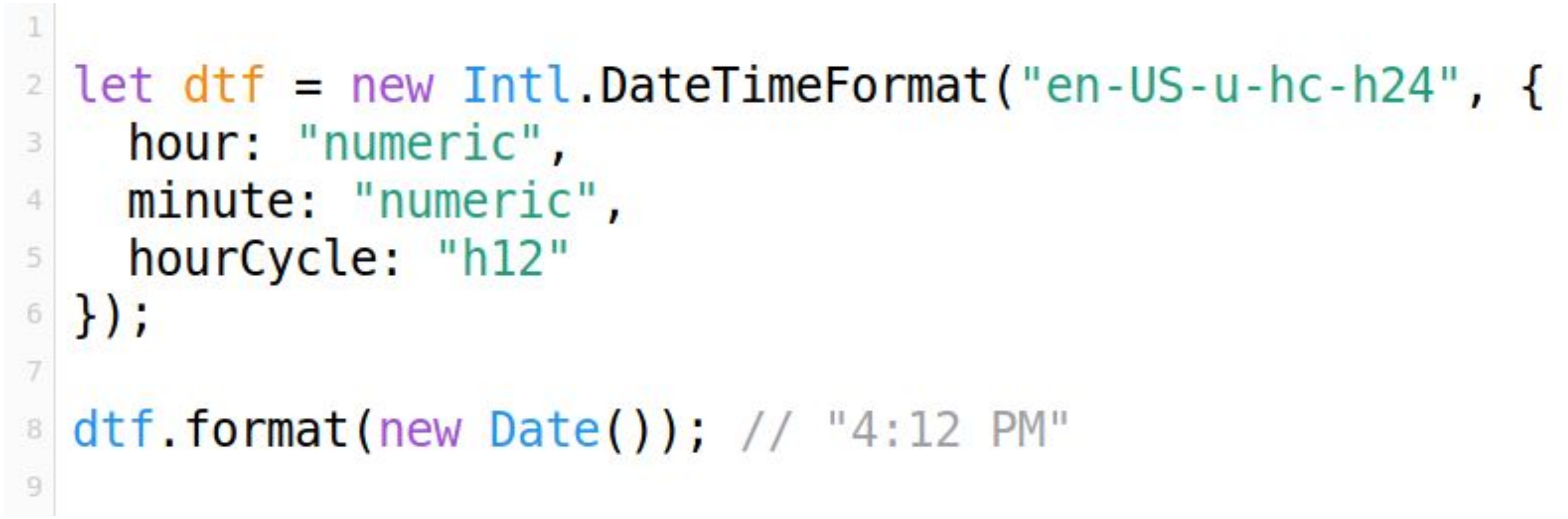
{ hourCycle: "h11" | "h12" | "h23" | "h24" }

New Option

- normalize hour cycle handling
- close the gap between Unicode Extensions & Option Bag
- deprecates `hour12` option

{ hourCycle: "h11" | "h12" | "h23" | "h24" }

New Option



```javascript

let dtf = new Intl.DateTimeFormat("en-US-u-hc-h24", {
  hour: "numeric",
  minute: "numeric",
  hourCycle: "h12"
});

dtf.format(new Date()); // "4:12 PM"

```

Active Proposals

Intl.Locale

sr-SR-u-ca-buddhist

```
{
    script: “Cyrl”,
    hourCycle: “h24”,
}
```

sr-Cyrl-SR-u-ca-buddhist-hc-h24

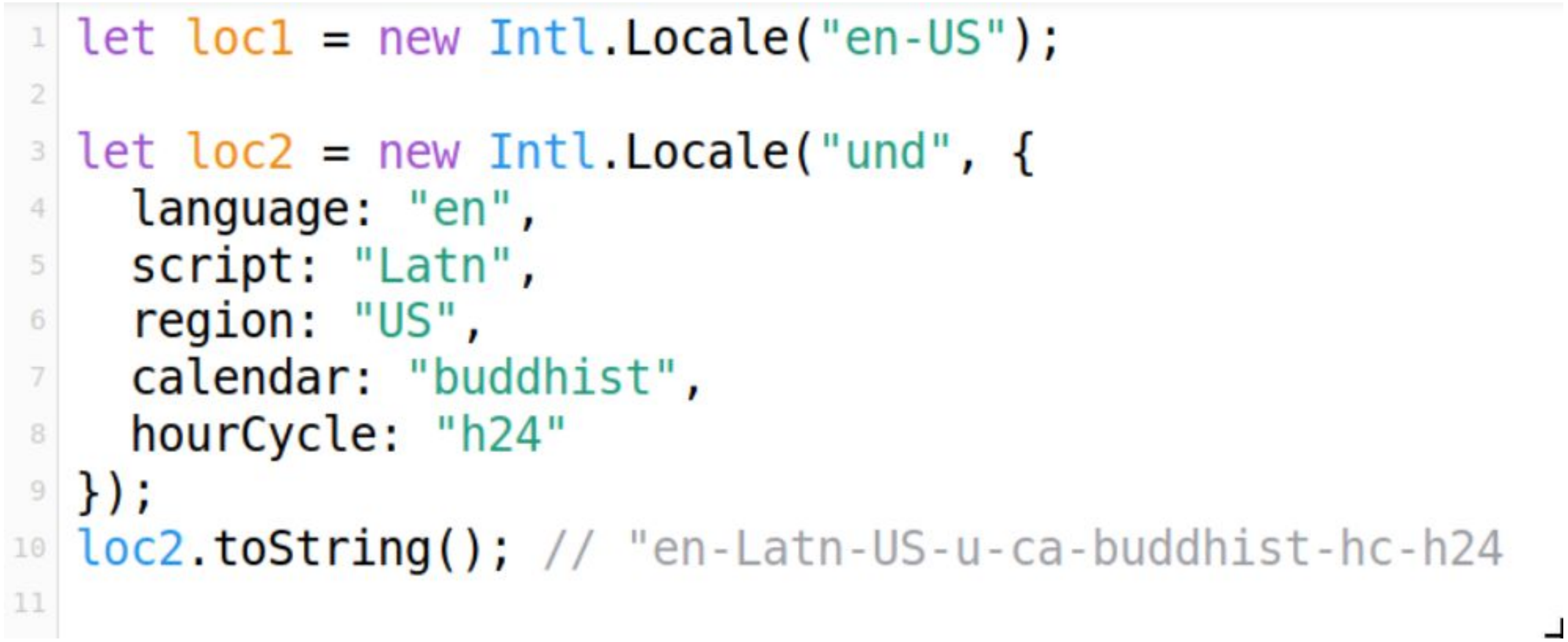
Intl.Locale

Goals

- Parse BCP47 language tags
- Manipulate Locale objects
- Serialize Locale objects to BCP47 language tags

Intl.Locale

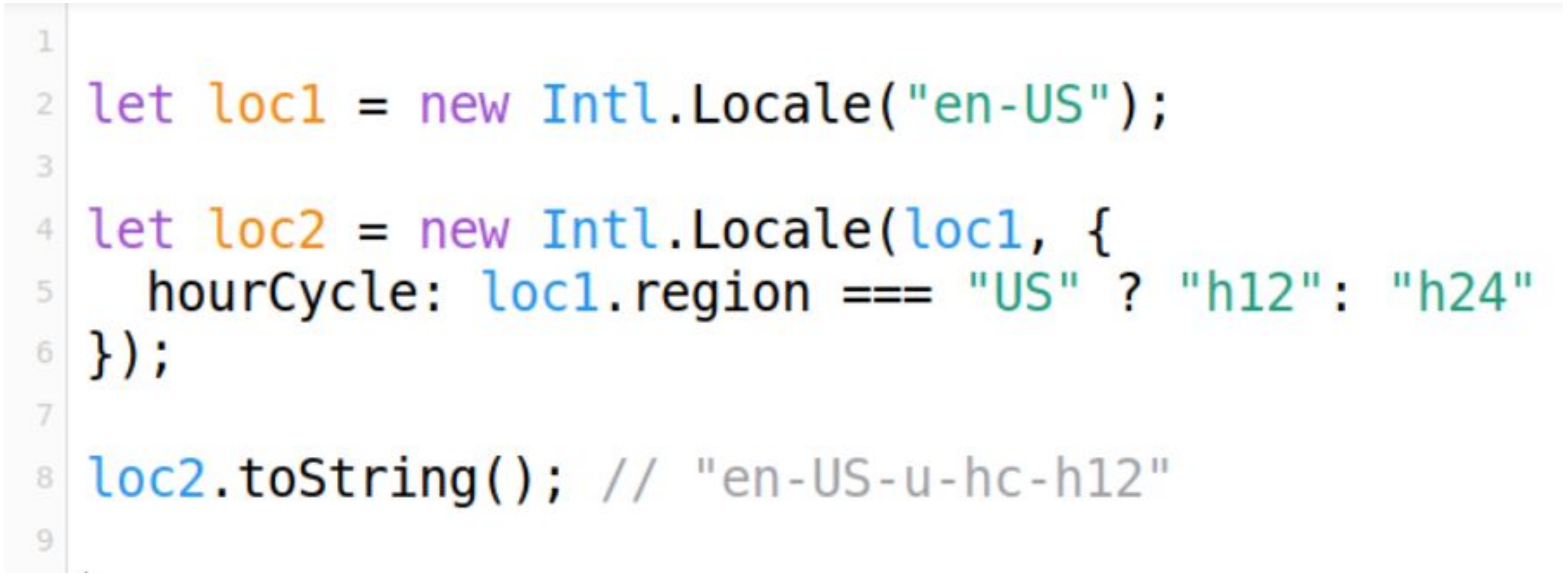
Parsing / Serializing



```
let loc1 = new Intl.Locale("en-US");

let loc2 = new Intl.Locale("und", {
  language: "en",
  script: "Latn",
  region: "US",
  calendar: "buddhist",
  hourCycle: "h24"
});
loc2.toString(); // "en-Latn-US-u-ca-buddhist-hc-h24
```

Intl.Locale

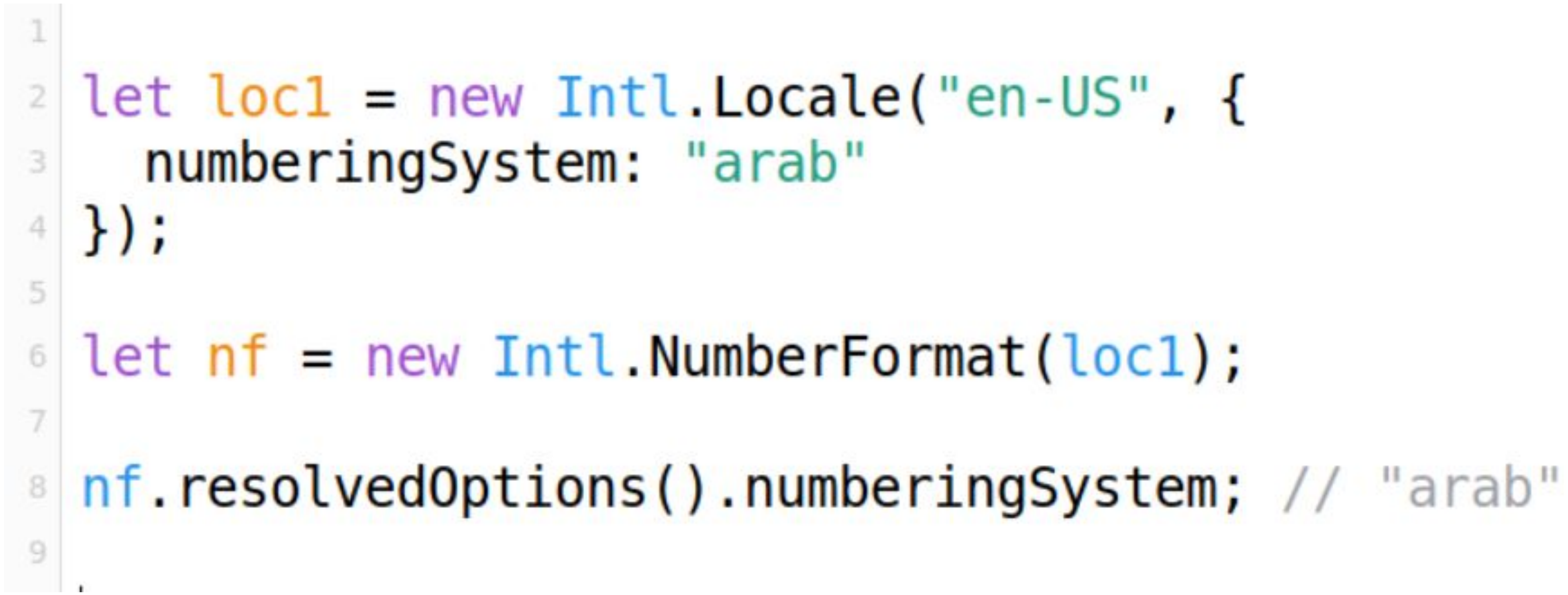
Manipulating



```
let loc1 = new Intl.Locale("en-US");

let loc2 = new Intl.Locale(loc1, {
  hourCycle: loc1.region === "US" ? "h12": "h24"
});

loc2.toString(); // "en-US-u-hc-h12"
```

Intl.Locale

Chaining



```
let loc1 = new Intl.Locale("en-US", {
  numberingSystem: "arab"
});

let nf = new Intl.NumberFormat(loc1);

nf.resolvedOptions().numberingSystem; // "arab"
```

Intl.ListFormat

Anna, John and Mary

5 minutes, 35 seconds

Poland, Germany or Slovakia

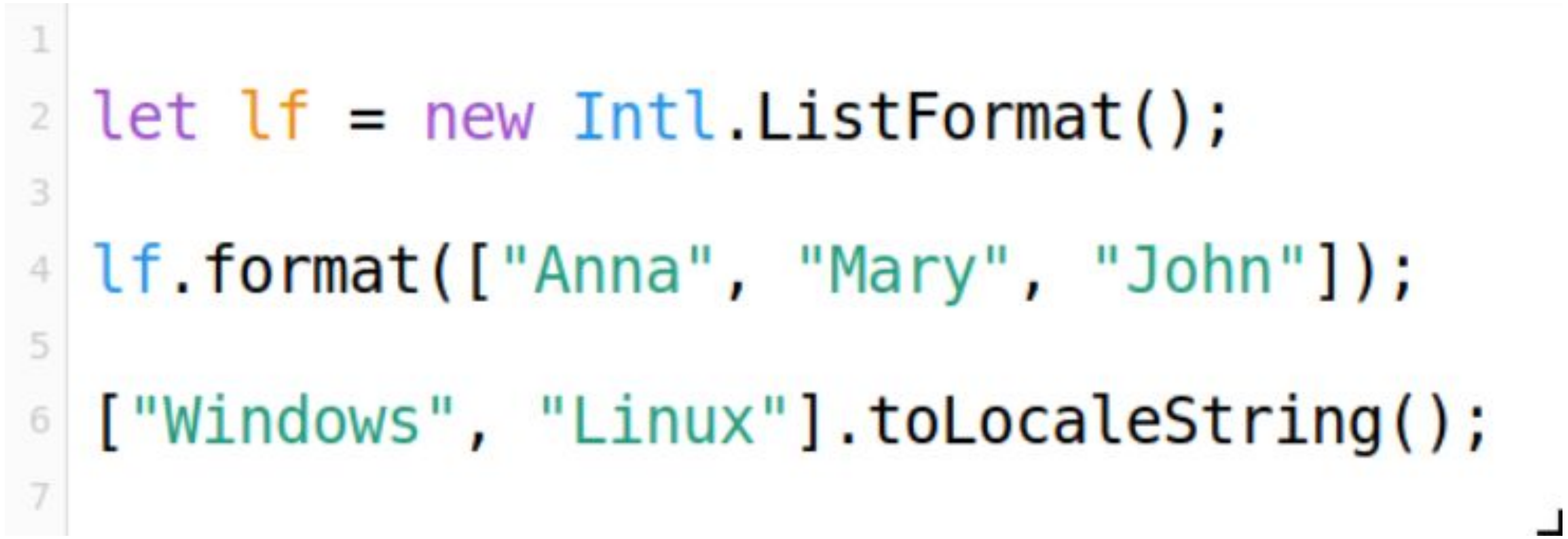
Intl.ListFormat

Goals

- Lay foundation for basic locale-aware list formatting
- Enable future combinations with other formatters
- Support core multilingual list options

Intl.ListFormat

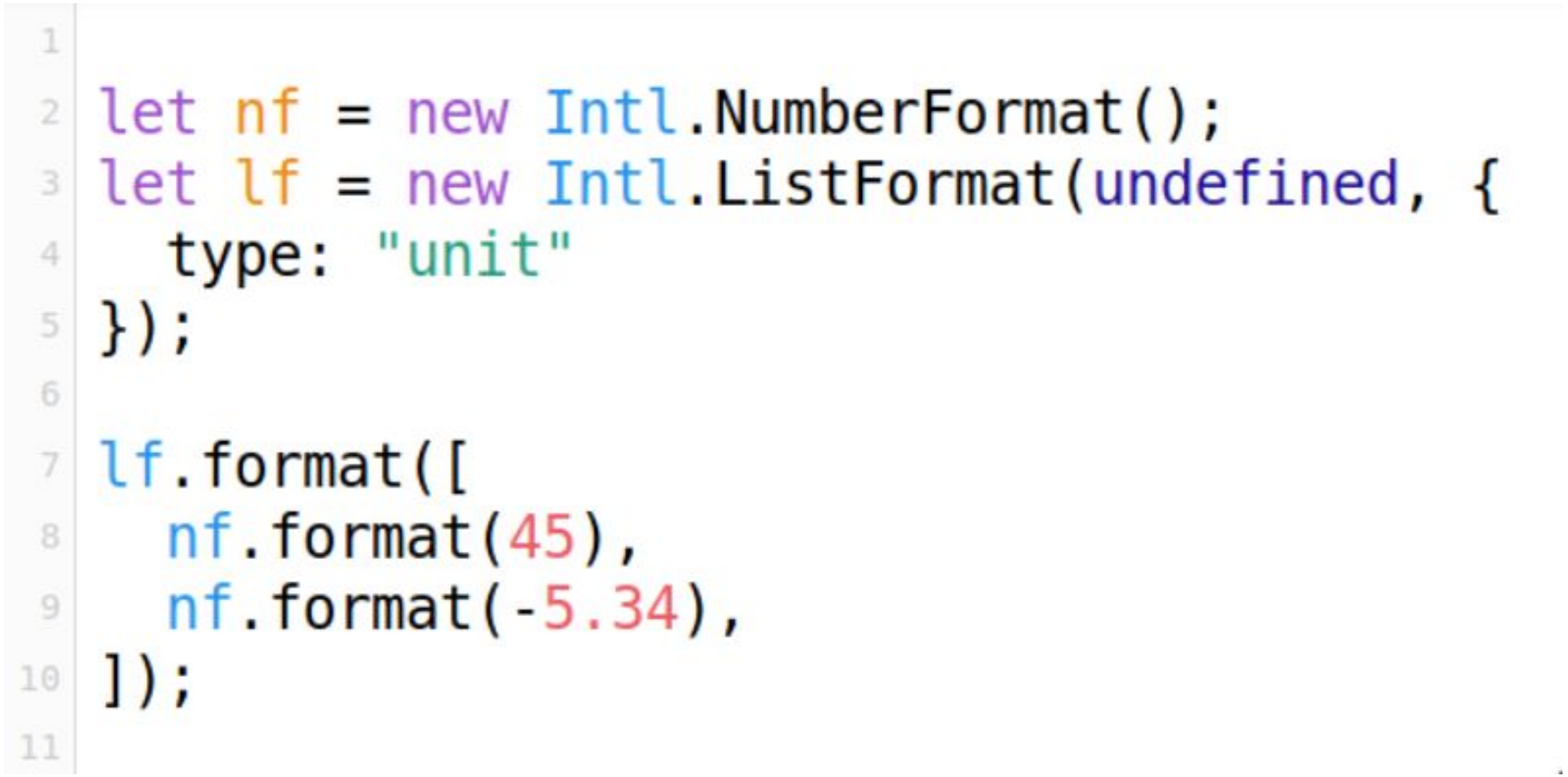
Basic use



```

let lf = new Intl.ListFormat();

lf.format(["Anna", "Mary", "John"]);

["Windows", "Linux"].toLocaleString();

```

Intl.ListFormat

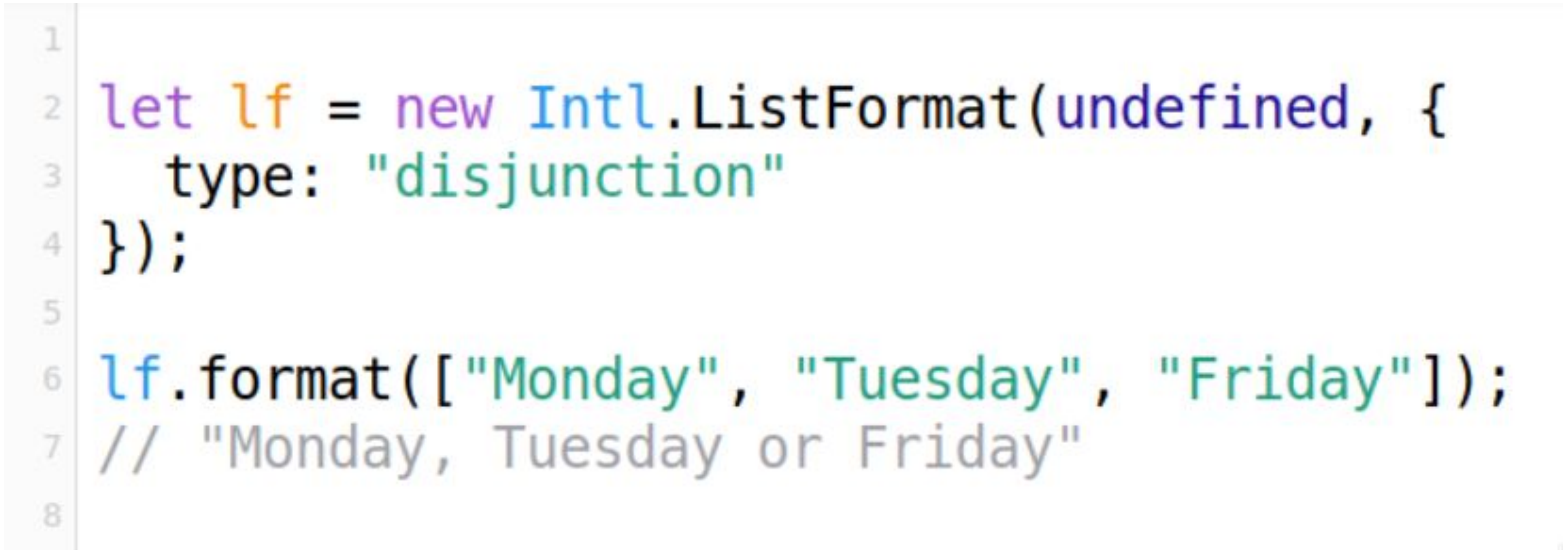
Combinations



```
let nf = new Intl.NumberFormat();
let lf = new Intl.ListFormat(undefined, {
  type: "unit"
});

lf.format([
  nf.format(45),
  nf.format(-5.34),
]);
```

Intl.ListFormat

Types



```
let lf = new Intl.ListFormat(undefined, {
  type: "disjunction"
});

lf.format(["Monday", "Tuesday", "Friday"]);
// "Monday, Tuesday or Friday"
```

Intl.NumberFormat 2

984.2 km/h

987.7 million

287.7E6

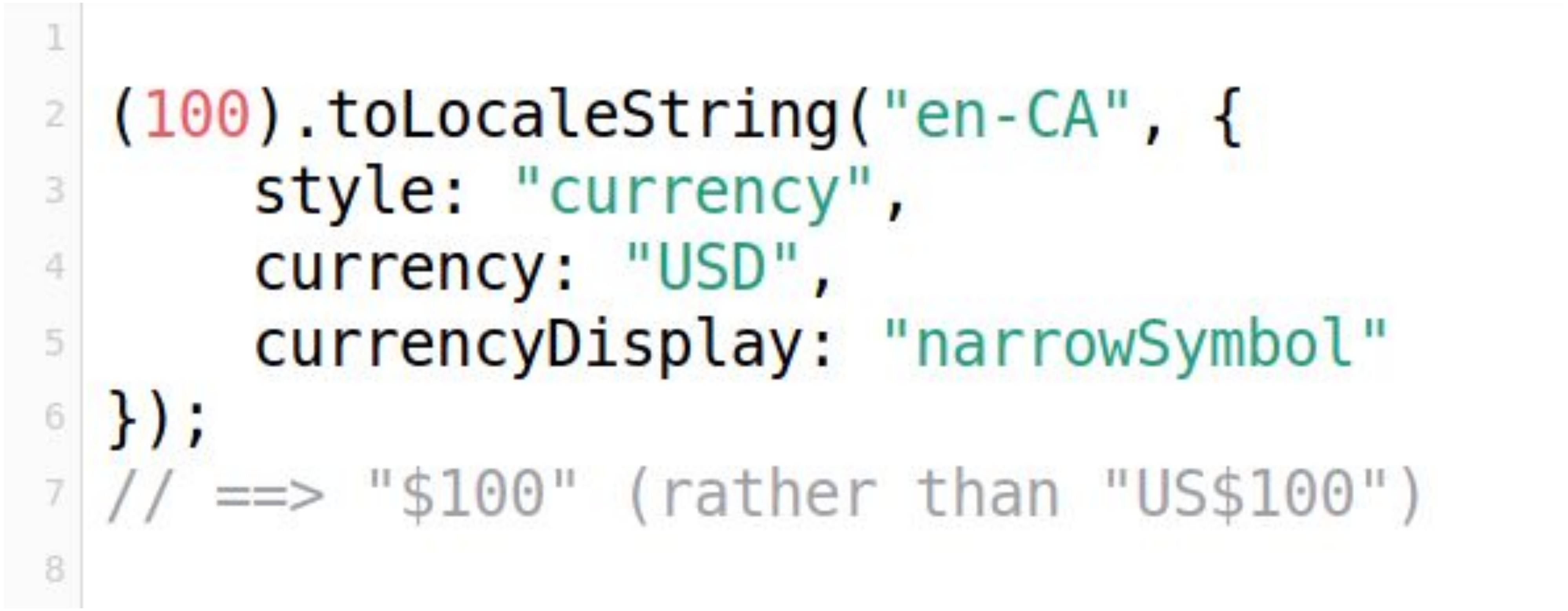
Intl.NumberFormat rev. 2

Goals

- Add measure unit formatting
- Add scientific and compat notations
- Add sign display

Intl.NumberFormat rev. 2

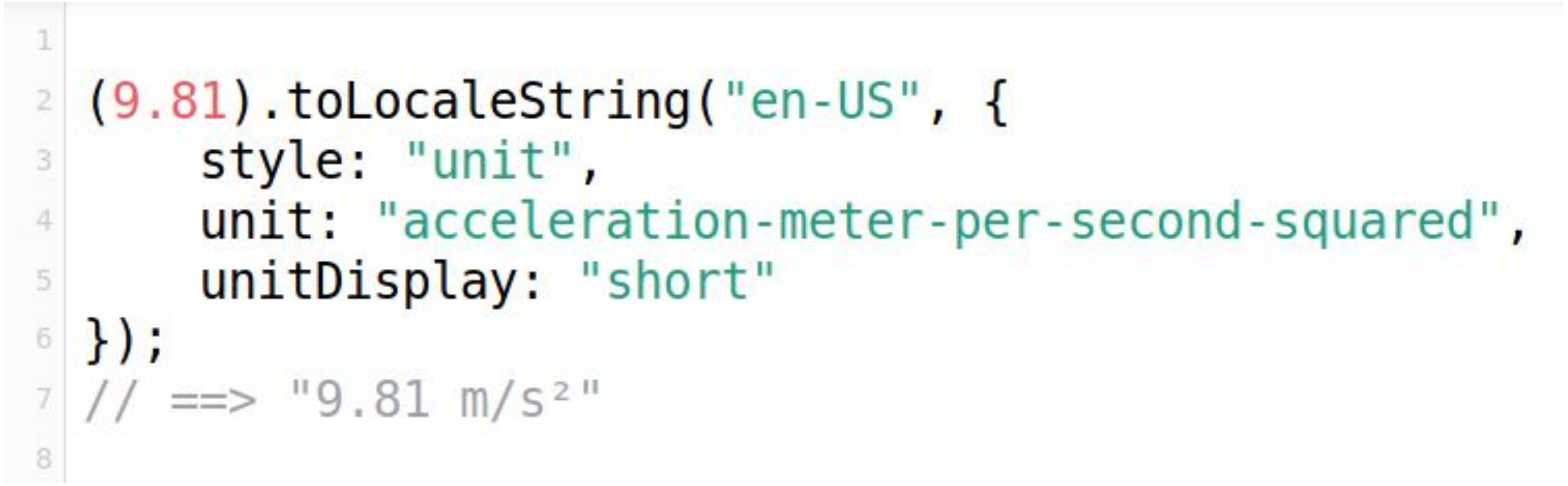
Narrow symbol currency



```

(100).toLocaleString("en-CA", {
    style: "currency",
    currency: "USD",
    currencyDisplay: "narrowSymbol"
});
// ==> "$100" (rather than "US$100")

```

Intl.NumberFormat rev. 2

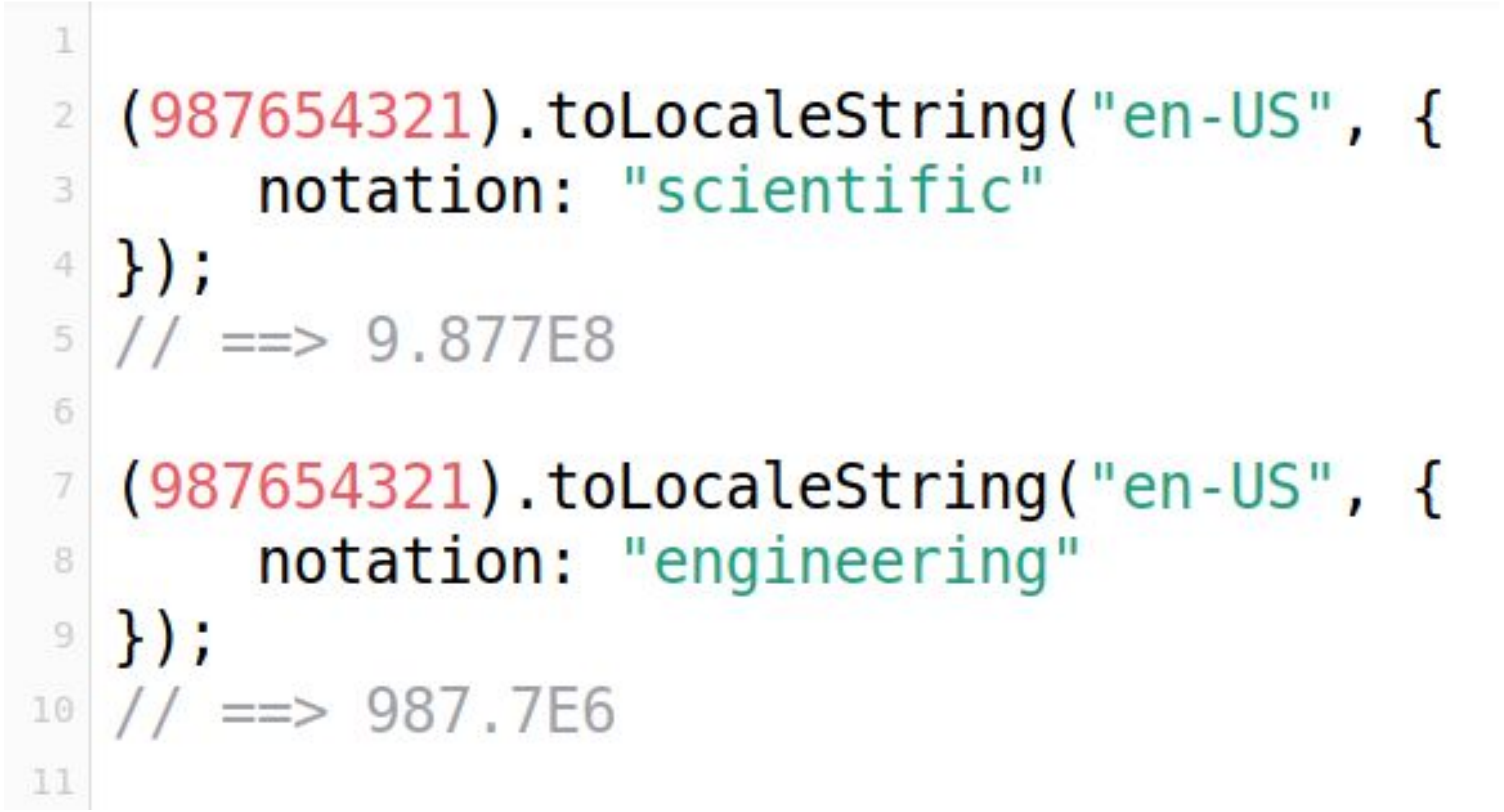
Units



```
(9.81).toLocaleString("en-US", {
    style: "unit",
    unit: "acceleration-meter-per-second-squared",
    unitDisplay: "short"
});
// ==> "9.81 m/s²"
```

Intl.NumberFormat rev. 2

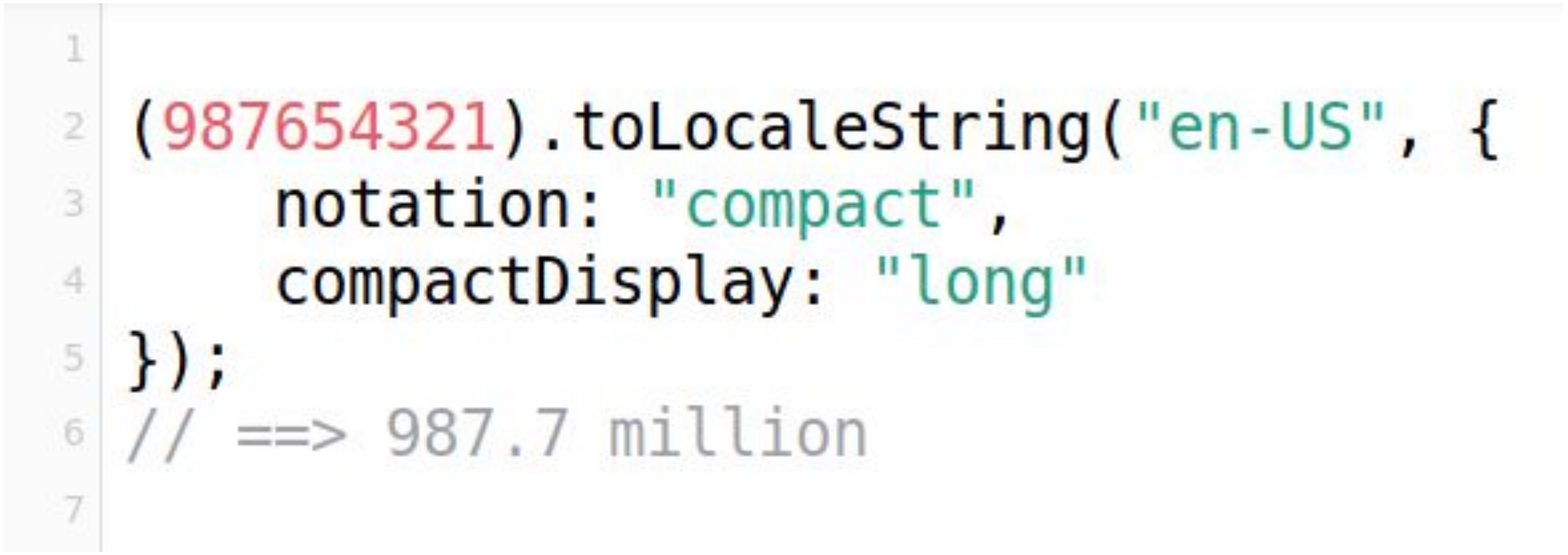
Scientific Notation



```
(987654321).toLocaleString("en-US", {
    notation: "scientific"
});
// ==> 9.877E8

(987654321).toLocaleString("en-US", {
    notation: "engineering"
});
// ==> 987.7E6
```

Intl.NumberFormat rev. 2

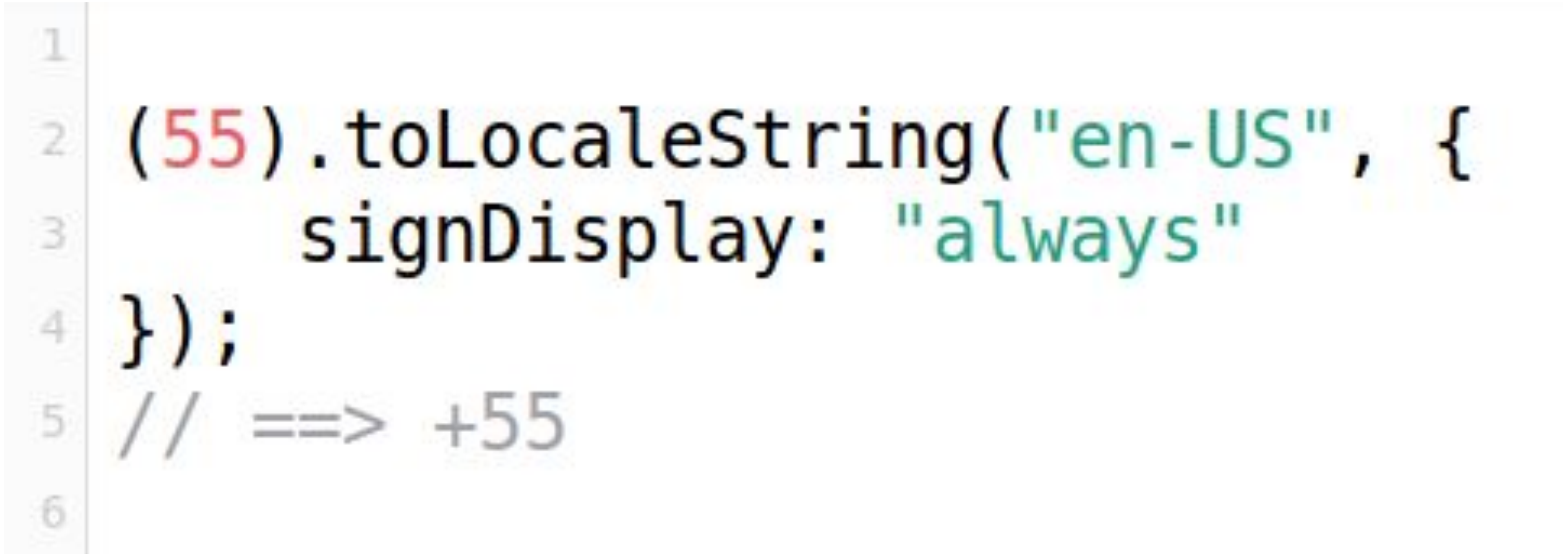
Compact Notation



```
(987654321).toLocaleString("en-US", {
    notation: "compact",
    compactDisplay: "long"
});
// ==> 987.7 million
```

Intl.NumberFormat rev. 2

Sign Display



```

(55).toLocaleString("en-US", {
    signDisplay: "always"
});
// ==> +55

```

Intl.
RelativeTimeFormat

in 5 minutes

2 yrs. ago

tomorrow

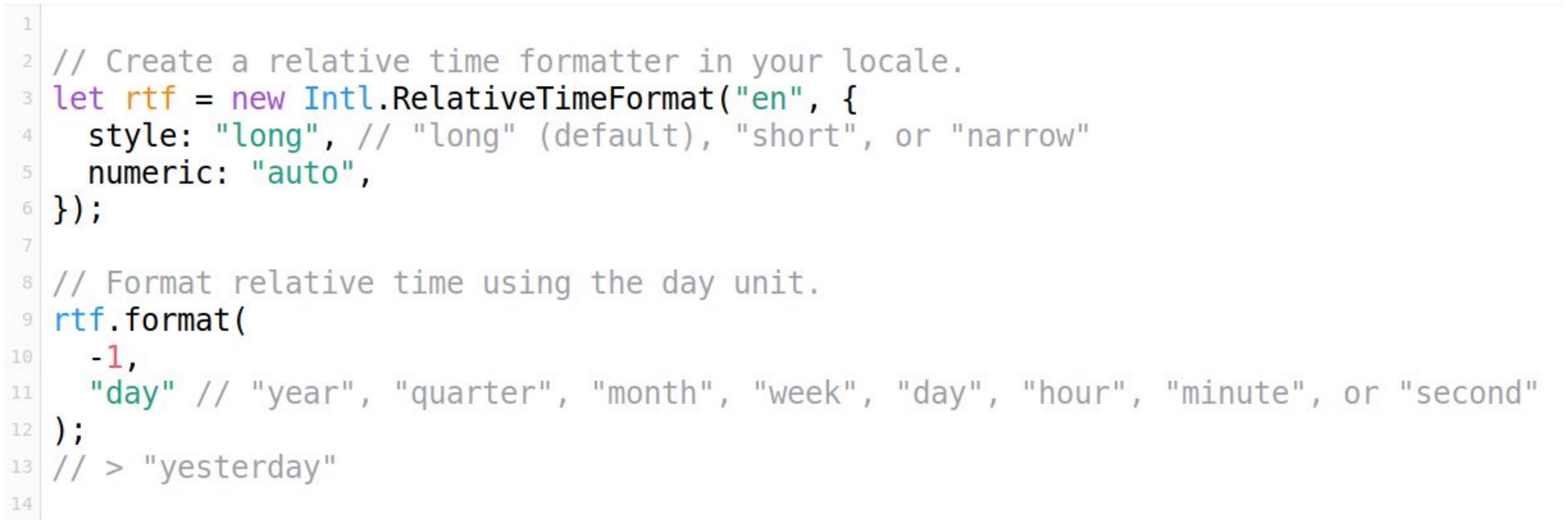
Intl.RelativeTimeFormat

Goals

- Add human readable relative time data
- Lower entry barrier for high level time APIs
- Enable locale-specific named relative time units

Intl.RelativeTimeFormat

Basic use



```

// Create a relative time formatter in your locale.
let rtf = new Intl.RelativeTimeFormat("en", {
  style: "long", // "long" (default), "short", or "narrow"
  numeric: "auto",
});

// Format relative time using the day unit.
rtf.format(
  -1,
  "day" // "year", "quarter", "month", "week", "day", "hour", "minute", or "second"
);
// > "yesterday"

```

Others…

21 - 27 July 2018

dateStyle / timeStyle

Intl.Segmenter

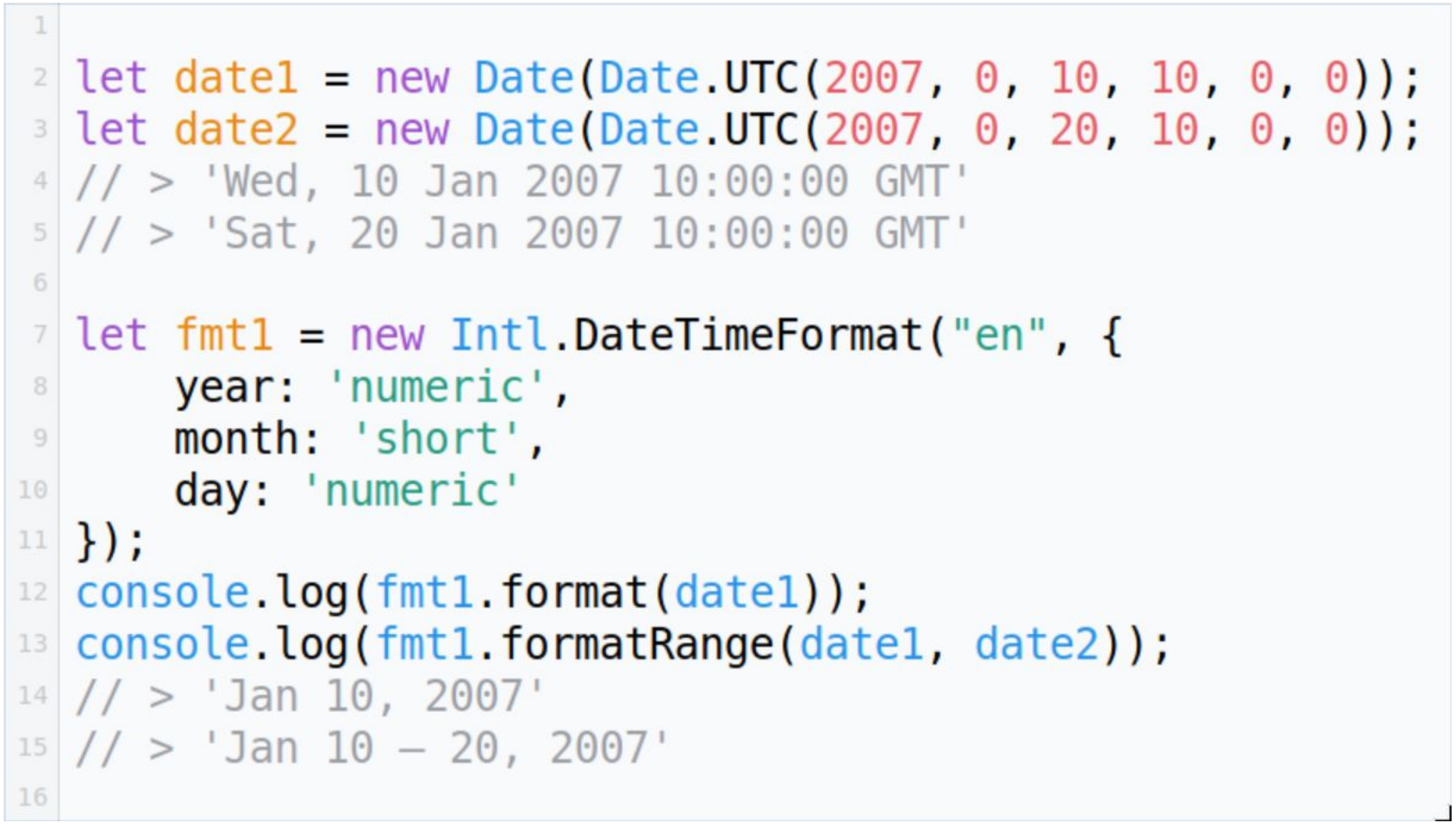
Others...

Goals

- Intl.DateTimeFormat.formatRange
- Intl.DateTimeFormat dateStyle/timeStyle
- Intl.Segmenter

Intl.DateTimeFormat.formatRange

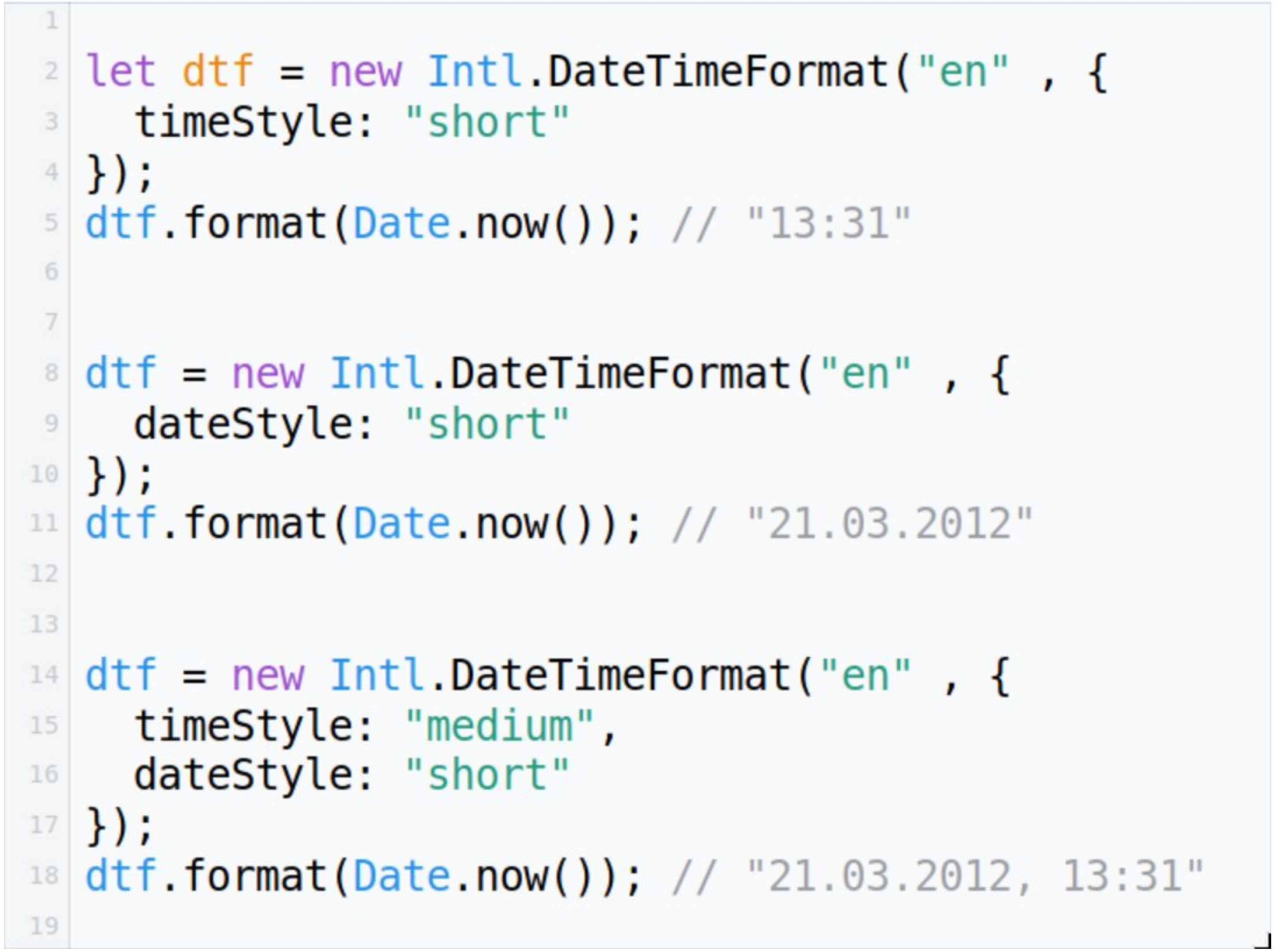
Basic use



```

let date1 = new Date(Date.UTC(2007, 0, 10, 10, 0, 0));
let date2 = new Date(Date.UTC(2007, 0, 20, 10, 0, 0));
// > 'Wed, 10 Jan 2007 10:00:00 GMT'
// > 'Sat, 20 Jan 2007 10:00:00 GMT'

let fmt1 = new Intl.DateTimeFormat("en", {
    year: 'numeric',
    month: 'short',
    day: 'numeric'
});
console.log(fmt1.format(date1));
console.log(fmt1.formatRange(date1, date2));
// > 'Jan 10, 2007'
// > 'Jan 10 – 20, 2007'

```

{ dateStyle, timeStyle }

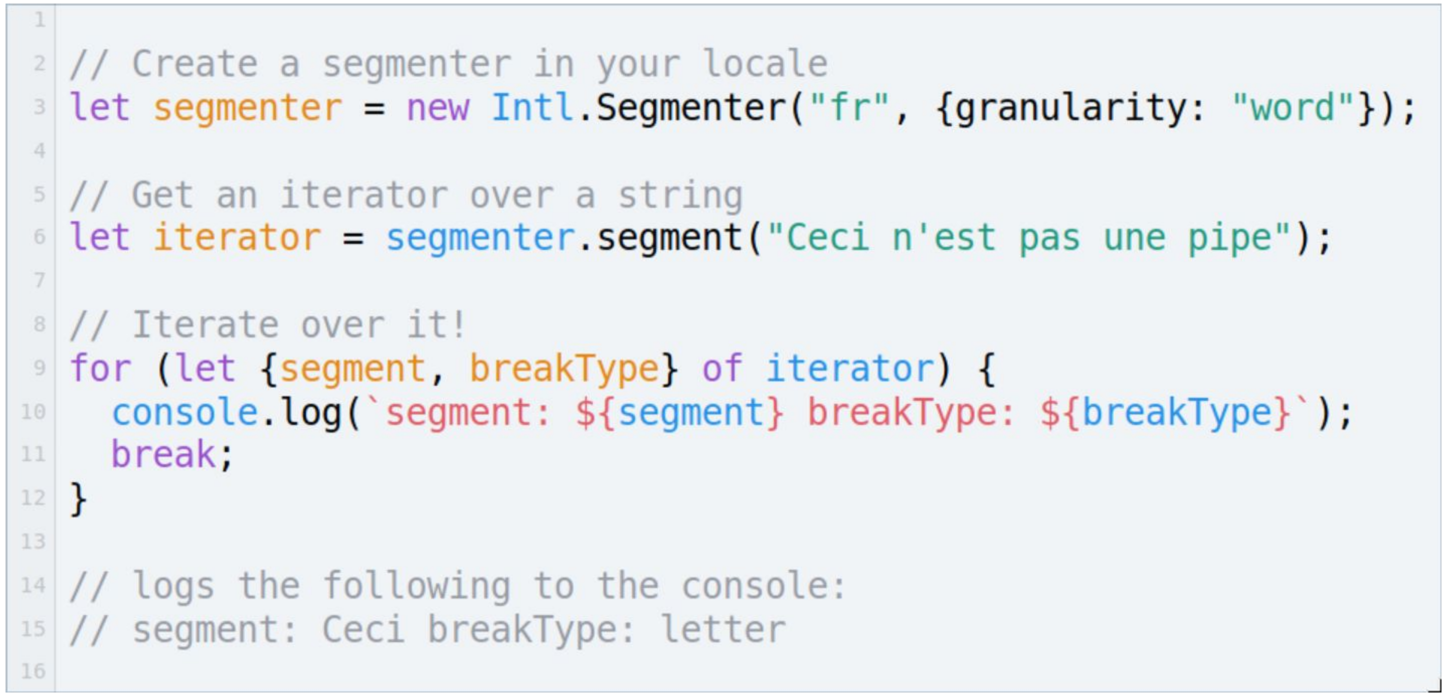
Basic use



```
let dtf = new Intl.DateTimeFormat("en" , {
  timeStyle: "short"
});
dtf.format(Date.now()); // "13:31"


dtf = new Intl.DateTimeFormat("en" , {
  dateStyle: "short"
});
dtf.format(Date.now()); // "21.03.2012"


dtf = new Intl.DateTimeFormat("en" , {
  timeStyle: "medium",
  dateStyle: "short"
});
dtf.format(Date.now()); // "21.03.2012, 13:31"

```

Intl.Segmenter

Basic use



```javascript

// Create a segmenter in your locale
let segmenter = new Intl.Segmenter("fr", {granularity: "word"});

// Get an iterator over a string
let iterator = segmenter.segment("Ceci n'est pas une pipe");

// Iterate over it!
for (let {segment, breakType} of iterator) {
  console.log(`segment: ${segment} breakType: ${breakType}`);
  break;
}

// logs the following to the console:
// segment: Ceci breakType: letter

```

How You Can Help

1. Test the APIs
2. Join the Conversation
3. File requests
4. Evangelize the Spec

https://github.com/tc39/ecma402/

Q&A

Please, take a moment to review this talk:

http://www.unicodeconference.org/eval-sp

TC
39

Thank You