Learning from the Mistakes of a v1

Sarah Dayan
Software Engineer, Algolia

@frontstuff_io

```
const price = Dinero({
  amount: 5000,
  currency: 'EUR'
});

const tip = price.percentage(20);
const fullPrice = price.add(tip);

fullPrice.toFormat(); // "€60.00"
```

2
years

400
commits

29
releases

@frontstuff_io

A v2 usually marks a turning point.

dinerojs / **dinero.js**

An immutable JavaScript library to create, calculate and format money.

dinerojs.com/

MIT License

3.7k stars

113 forks

It uses numbers to represent amounts

```
const price = Dinero({
  amount: 5000,
  currency: 'EUR'
});
```

```
.1 + .2;

// 0.30000000000000004

(1 + 2) / 10

// 0.3
```

```
const price = Dinero({
  // 5000 cents, or 50 euros
  amount: 5000,
  currency: 'EUR'
});
```

```
Number.MAX_SAFE_INTEGER;

// 9007199254740991

Number.MIN_SAFE_INTEGER;

// -9007199254740991
```

Enter bigints.

```
const calculator = {
  add(a, b) {
    return a + b;
  },
  subtract(a, b) {
    return a - b;
  },
  // ...
};
```

```
import { Decimal } from 'decimal.js';

const bigDinero = createDinero({
  calculator: {
    add: Decimal.add,
    subtract: Decimal.sub,
    // ...
  },
});

const price = bigDinero({
  amount: new Decimal(Number.MAX_SAFE_INTEGER),
  currency: 'EUR',
});
```

It doesn't handle different exponents

1 euro = 100 cents

1 x 10^2 = 100

```
const price = Dinero({
  amount: 5000,
  currency: 'EUR',
});

price.toFormat(); // "€50.00"
```

```
const price = Dinero({
  amount: 1000,
  currency: 'IQD',
  precision: 3,
});
```

```
const IQD = {
  code: 'IQD',
  base: 10,
  exponent: 3,
};

const price = Dinero({
  amount: 1000,
  currency: IQD,
});
```

It expects a REST API for conversions

```
const options = {
  endpoint: 'https://api/?base={{from}}',
  propertyPath: 'data.rates.{{to}}',
  headers: {
    'user-key': 'xxxxxxxxxx',
  },
  roundingMode: 'HALF_UP',
};

Dinero({
  amount: 5000,
  currency: 'EUR',
}).convert('XBT', options);
```

```
const price = Dinero({
  amount: 5000,
  currency: 'EUR',
});

price.convert('XBT');
```

```
const rates = new Promise((resolve) =>
  resolve({
    XBT: 0.00012,
  })
);

Dinero({ amount: 5000 })
  .convert(EUR, { rates });
```

```
const rates = {
  rates: {
    XBT: 0.00012,
  },
};

Dinero({
  amount: 5000,
  currency: 'EUR',
}).convert('XBT', {
  endpoint: new Promise((resolve) =>
    resolve(rates)
  ),
});
```

It relies on the ECMAScript I18n API

```
const amount = 5;

amount.toLocaleString('en-GB', {
  style: 'currency',
  currency: 'EUR',
});

// "€5.00"
```

```
const options = {
  style: 'currency',
  currency: 'EUR',
};

amount.toLocaleString('en-GB', options);

// "€5.00"

amount.toLocaleString('de-DE', options);

// "5,00 €"
```

```
const price = Dinero({
  amount: 500,
  currency: 'EUR',
}).setLocale('en-GB');

price.toFormat();

// "€5.00"
```

Tight coupling

Lack of flexibility

Inconsistent implementation

```
const price = Dinero({
  amount: 500,
  currency: EUR,
});

price.toFormat(
  ({ amount, currency }) =>
    `${currency.code} ${amount}`,
  {
    digits: 2,
    roundingMode: Math.round,
  }
);

// "EUR 5"
```

```
function i18n({ amount, currency }) {
  return amount.toLocaleString('en-GB', {
    style: 'currency',
    currency: currency.code,
  });
}

price.toFormat(i18n, {
  digits: 2,
  roundingMode: Math.round,
});

// "€5.00"
```

```
const currencySymbols = {
  EUR: '€',
  USD: '$',
  // ...
};

function manualI18n({ amount, currency }) {
  const { code } = currency;
  const symbol = currencySymbols[code] || code;

  return `${symbol}${amount.toFixed(2)}`;
}

price.toFormat(manualI18n, {
  digits: 2,
  roundingMode: Math.round,
});

// "€5.00"
```

You can't tree-shake it

dinero.js@1.8.1

tree-shakeable | no dependencies

BUNDLE SIZE

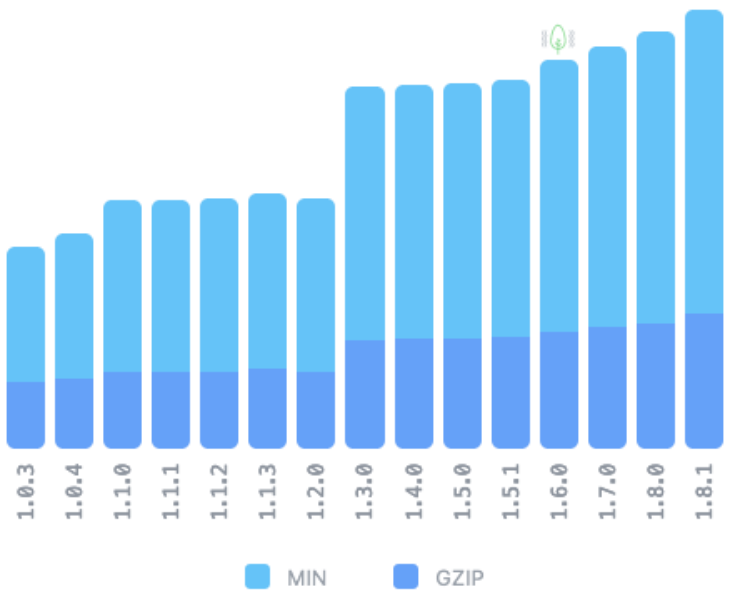
9.9 kB
MINIFIED

3.1 kB
MINIFIED + GZIPPED

DOWNLOAD TIME

103 ms
2G EDGE

62 ms
EMERGING 3G



```
const price = Dinero({
  amount: 500,
  currency: 'EUR',
});

price.add(price).multiply(4);
```

```
const price = Dinero({
  amount: 500,
  currency: 'EUR',
});

multiply(add(price, price), 4);
```


WHY NOT BOTH?

It's written in JavaScript

Maintaining types by hand doesn't scale.

And it's a pain.

Static typing is extremely helpful.

TypeScript hits two birds with one stone.

What happened?

First library

No real-world experience

Many assumptions

Scalability
Flexibility
Agnosticism
DX

Thank you!

Sarah Dayan

Software Engineer, Algolia

Links

Why Javascript Numbers Are Weird (And How to Fix It)

youtu.be/Wq2AqONg7rs

Dinero.js on GitHub

github.com/dinerojs


jmbr
software


SECURE CODE
WARRIOR


howest
university of applied sciences


algolia


DataCamp


Microsoft


XM CYBER


NOKIA Bell Labs


PROCORE


THREADS


JS


DigitalOcean


Vue Mastery


JET
BRAINS