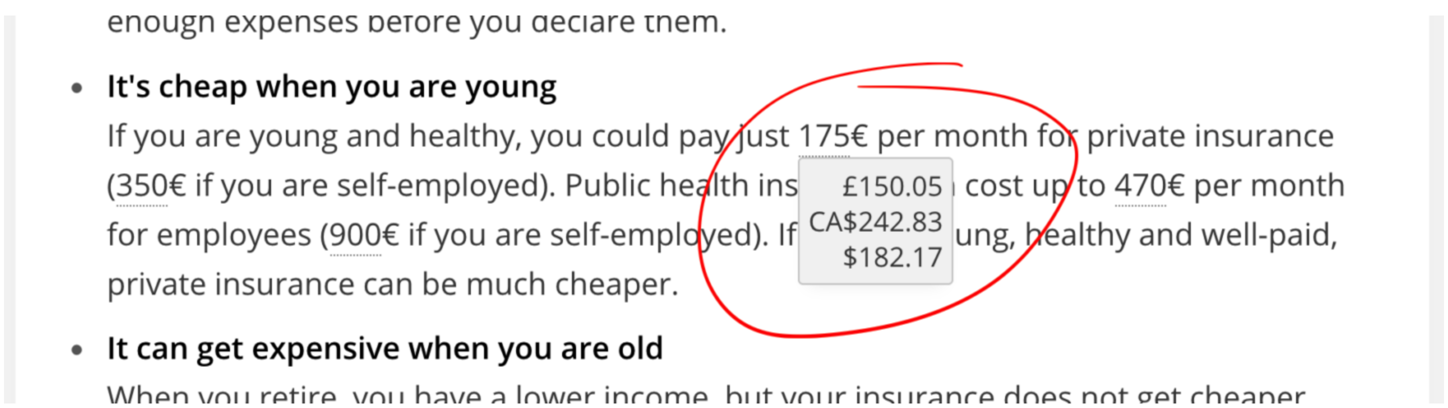
How I built currency conversion tooltips

Posted on November 24, 2022

When I'm abroad, I struggle to make sense of amounts in other currencies.

This is also a problem for [All About Berlin]'s visitors. 40% of them are outside of Germany, and a chunk of the remaining 60% are visitors and recent immigrants.

I wanted to express numbers in currencies that they are familiar with, so **I built an inline currency conversion tooltip**. [See it in action here].


enough expenses before you declare them.

- **It's cheap when you are young**
  If you are young and healthy, you could pay just 175€ per month for private insurance (350€ if you are self-employed). Public health ins cost up to 470€ per month for employees (900€ if you are self-employed). If ung, healthy and well-paid, private insurance can be much cheaper.

£150.05
CA$242.83
$182.17

- **It can get expensive when you are old**
  When you retire, you have a lower income, but your insurance does not get cheaper

Table of contents

I get my exchange rates from the Open Exchange Rate API. Their free plan allows 1,000 API calls per month, or about one call every 45 minutes. I have far more than one visitor every 45 minutes, so I must cache the exchange rates.

I used the same approach as for the Plausible tracking script: proxy the request through All About Berlin's server, and cache the response for a few hours. Calling `allaboutberlin.com/api/exchangerates.json` returns a cached version of `openexchangerates.org/api/latest.json`.

Here's the relevant nginx config:

```
location = /api/exchangerates.json {
    proxy_pass https://openexchangerates.org/api/latest.json?app_id=.
    proxy_cache jscache;

    # Save valid responses for 6 hours. Serve latest valid response i
    proxy_cache_valid 200 6h;
    proxy_cache_use_stale updating error timeout invalid_header http_

    # Avoid SSL errors
    proxy_set_header Host openexchangerates.org;
    proxy_ssl_name openexchangerates.org;
    proxy_ssl_server_name on;
    proxy_ssl_session_reuse off;

    # Avoid caching the same page multiple times due to query params
    proxy_cache_key "openexchangerates";
}
```

The displayed exchange rates will be 6 hours old at most. If that's not good enough, you have a far greater problem. When the cached response is too old, the next request fetches the latest version, and it's cached for another 6 hours.

Proxying external API requests through my server has another benefit: all requests are made to the same domain. This saves DNS query and speeds things up.

This is not a perfect approach, because every page now makes an extra request to get the exchange rates. I wanted to embed the exchange rates in the page before serving it, but

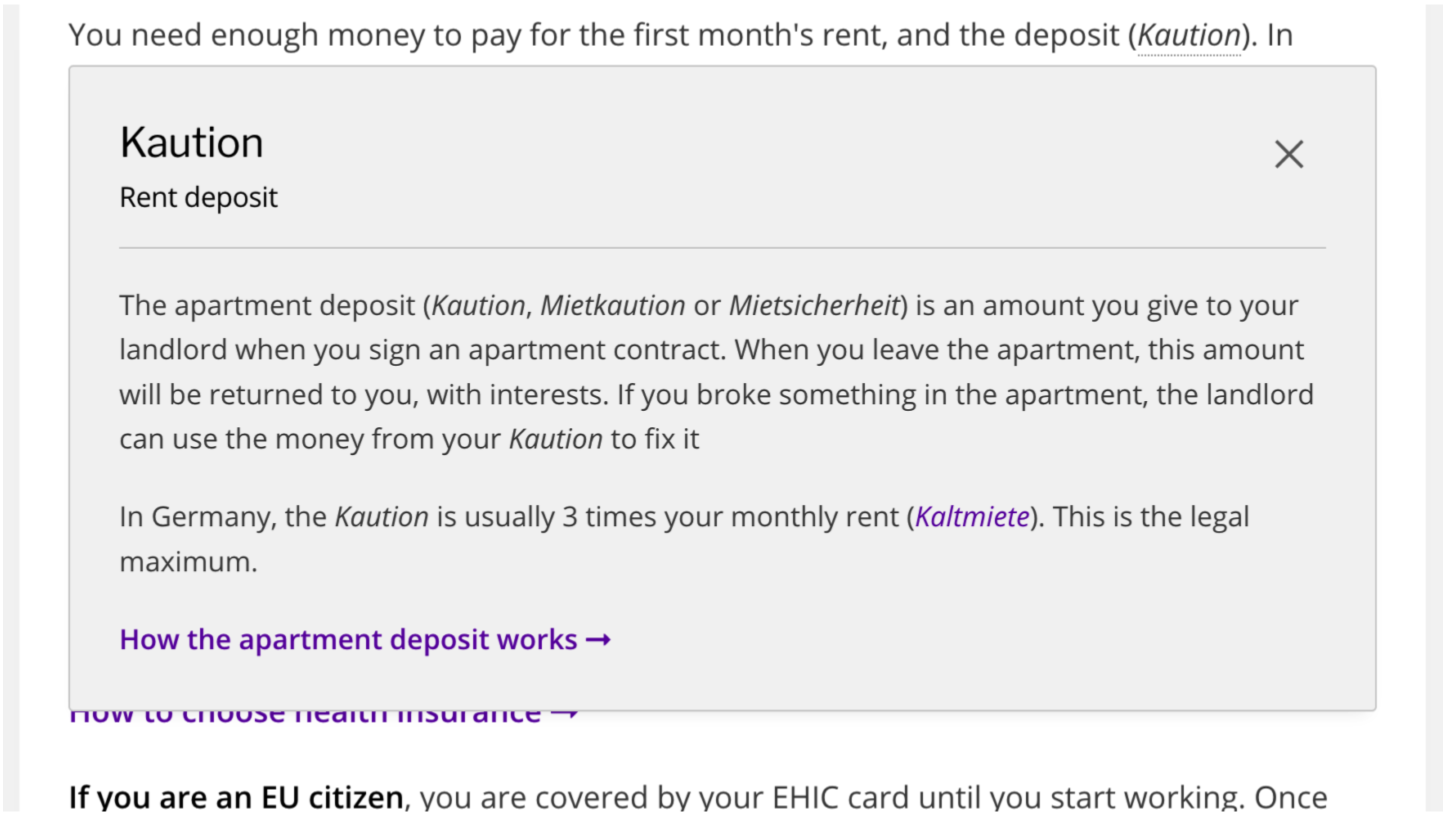
but update the exchange rates every few hours.

Embedding the exchange rates in the page would also be a problem for The Internet Archive. The old copies of a page should not show old exchange rates.

In the end, an extra request is not such a big deal. It has no perceptible impact, because readers only use those tooltips after a few seconds of reading.

Building the tooltips

Glossary tooltips were a pain to implement. It's surprisingly hard to build tooltips that open on hover if you have a mouse, and on tap if you have a touch screen.



You need enough money to pay for the first month's rent, and the deposit (*Kaution*). In

Kaution

Rent deposit

The apartment deposit (*Kaution*, *Mietkaution* or *Mietsicherheit*) is an amount you give to your landlord when you sign an apartment contract. When you leave the apartment, this amount will be returned to you, with interests. If you broke something in the apartment, the landlord can use the money from your *Kaution* to fix it

In Germany, the *Kaution* is usually 3 times your monthly rent (*Kaltmiete*). This is the legal maximum.

How the apartment deposit works →

How to choose health insurance →

If you are an EU citizen, you are covered by your EHIC card until you start working. Once

Glossary tooltips

Currency conversion tooltips were easier to build because they behave the same way on all devices. They close when you move your mouse or lift your finger. They don't need to stay open because they don't contain paragraphs of text.

When rendering the content, I wrap euro amounts in a `<span>` tag. `19€` becomes `<span class="currency">19</span>€`. This is done with the `replace` filter in twig.

When the page loads, I fetch the exchange rates, format them nicely with Intl.NumberFormat, and set the element's `data-currencies` attribute.

```
fetch('/api/exchangerates.json')
    .then(response => response.json())
    .then(data => {
        document.querySelectorAll('.currency').forEach(el => {
            // Convert from EUR to USD, and from USD to others
            const usdValue = Number(el.textContent.replace(',', ''))
            const asCurrency = curr => Intl.NumberFormat('en-US', {st
            el.dataset.currencies = `${asCurrency('USD')}\n${asCurren
        });
    });
```

This is the resulting HTML:

```
<span class="currency" data-currencies="$4,161\CA$5,555">3,994</span>
```

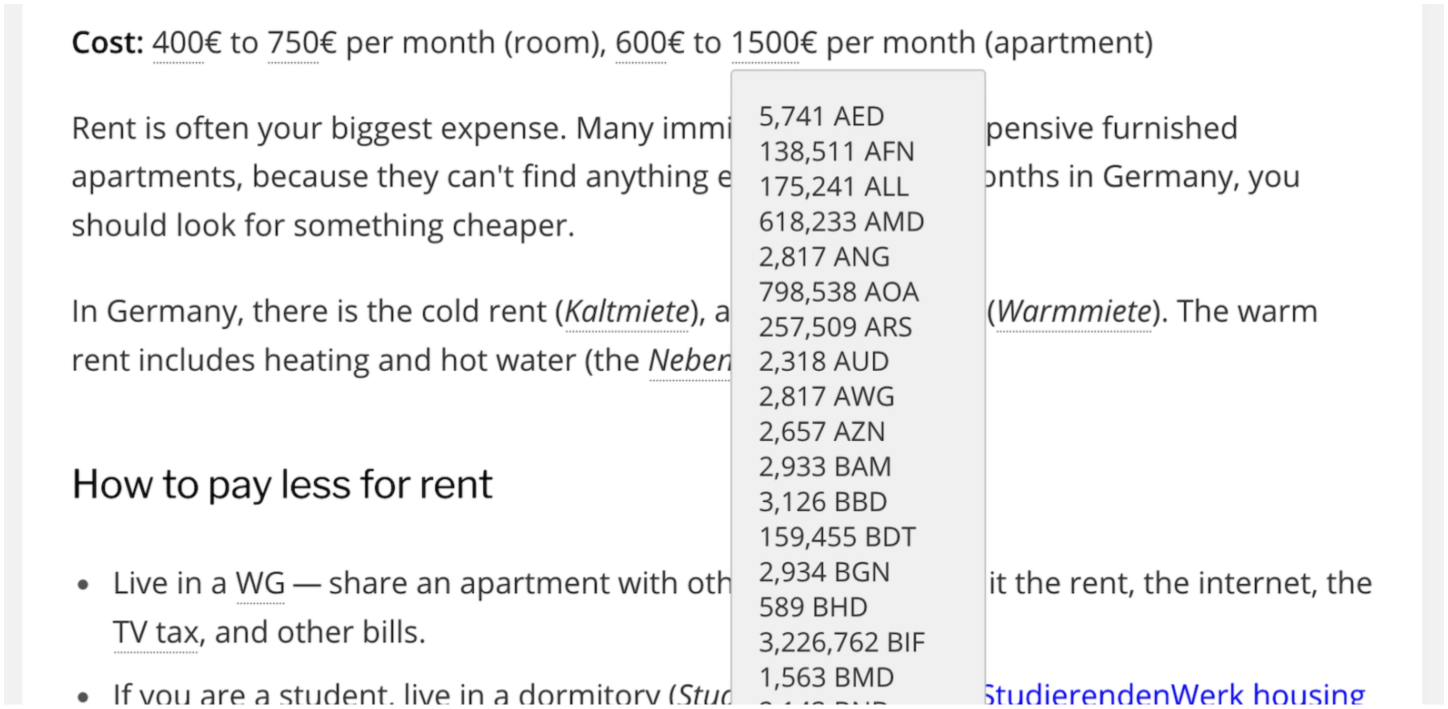
Next, I use a bit of CSS to display this information as a tooltip:

```
.currency[data-currencies]{
    color:inherit!important;
    border-bottom:1px dotted var(--color-border-02);
    cursor:help;
    position:relative;
}
.currency[data-currencies]:hover::after{
    content:attr(data-currencies);
    white-space:pre; /* Only break line on newline characters */
    display:block;
    position:absolute;
    top:var(--line-height);
    left:0;
    z-index:1000;

    font-size:var(--font-size-s);
    line-height:var(--line-height-compact);
    background:var(--color-background-widget);
    padding:var(--spacing-m);
    border:var(--border-widget);
    text-align:right;
}
```

Showing the right currencies

Open Exchange Rates lists 169 currencies. I can't show all of them.



Cost: 400€ to 750€ per month (room), 600€ to 1500€ per month (apartment)

Rent is often your biggest expense. Many immi pensive furnished apartments, because they can't find anything e onths in Germany, you should look for something cheaper.

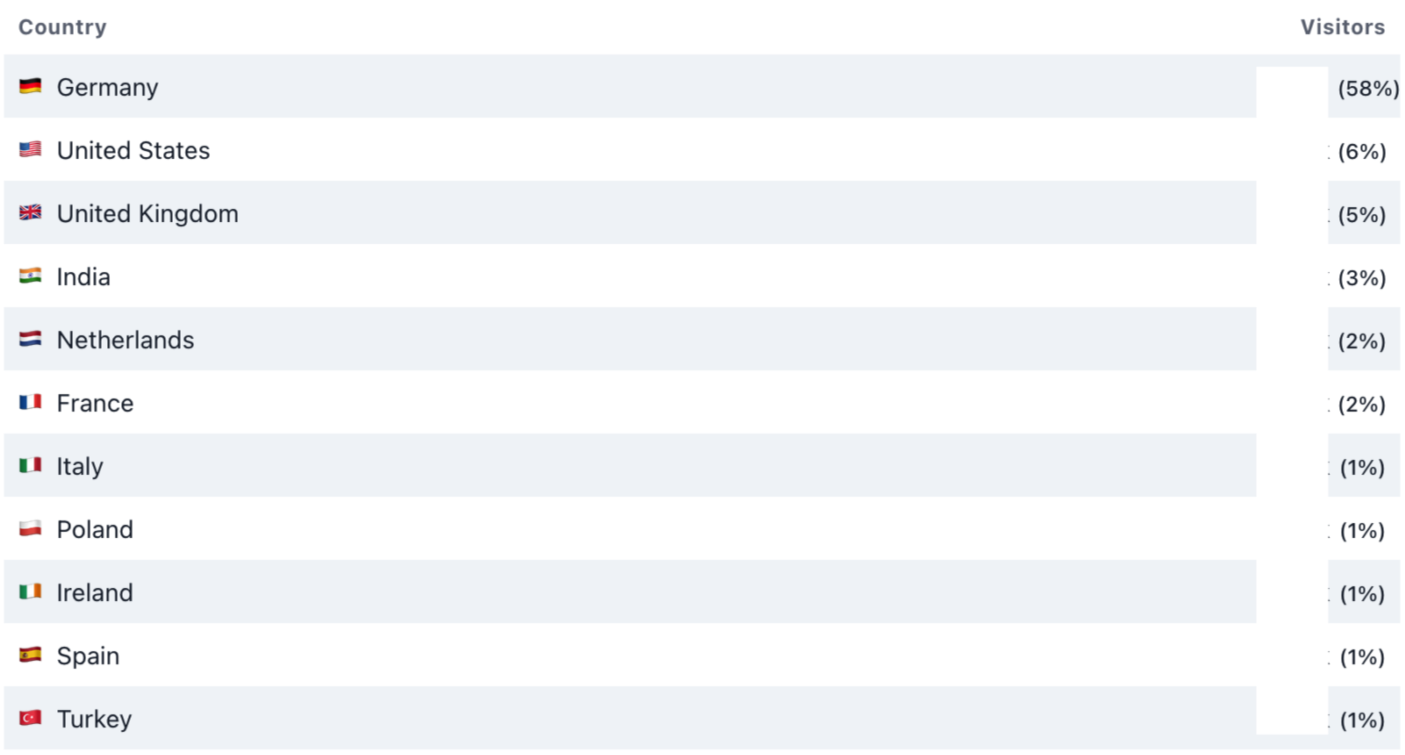
In Germany, there is the cold rent (*Kaltmiete*), a (*Warmmiete*). The warm rent includes heating and hot water (the *Neben*

How to pay less for rent

- Live in a WG — share an apartment with oth it the rent, the internet, the TV tax, and other bills.
- If you are a student, live in a dormitory (*Stud* StudierendenWerk housing

```
5,741 AED
138,511 AFN
175,241 ALL
618,233 AMD
2,817 ANG
798,538 AOA
257,509 ARS
2,318 AUD
2,817 AWG
2,657 AZN
2,933 BAM
3,126 BBD
159,455 BDT
2,934 BGN
589 BHD
3,226,762 BIF
1,563 BMD
```

I looked at All About Berlin's stats, and ranked my visitors by country. There were a few surprises there: Indian, Polish and Turkish visitors outrank those of a few anglophone countries. Never assume who your visitors are!



Country	Visitors
Germany	(58%)
United States	(6%)
United Kingdom	(5%)
India	(3%)
Netherlands	(2%)
France	(2%)
Italy	(1%)
Poland	(1%)
Ireland	(1%)
Spain	(1%)
Turkey	(1%)

I picked the 5 most popular currencies, but it left out 3 of Germany's neighbours. If I added more currencies, the tooltip became hard to read.

Instead, I check the user's preferred locales in `navigator.languages`, and use that to choose which currencies to list. If that doesn't work, I fall back to the hard-coded top 3.

I used country-json's data to make a map of country codes ("US") to currency codes ("USD"). You can see the code for it on Gist. I realised later that the data is wrong, and shows many outdated currencies.

Then, I used that map to show the right currencies:

```
fetch('/api/exchangerates.json')
    .then(...)
    .then(data => {
        ...
        const defaultCurrencyCodes = ["USD", "GBP", "INR", "PLN", "TR
        const countryCodeToCurrencyCode = {"AF":"AFN","AL":"ALL",...}
        const selectedCurrencyCodes = new Set(
            navigator.languages.map(l => countryCodeToCurrencyCode[l.
                .filter(Boolean) // Filter out empty country codes
                .concat(defaultCurrencyCodes) // Add the default coun
        ) // Add to a Set to only keep unique elements
        ...
    });
```

My preferred locales are de-DE, en-UK and fr-CA, so I see British pounds, Canadian dollars, and US dollars as a last default.

Graceful degradation and other improvements

I often have an unreliable internet connection on the U-Bahn, in hotels, in developing countries, and in Brandenburg. All About Berlin is just text on page, so it should be fully functional as soon as the text is visible on the page. This is why I make such a fuss about making extra requests.

The CSS rules above only apply if the `data-currencies` attributes is set. If the exchange rates have not loaded, the readers just see normal text.

I was also worried about showing outdated exchange rates. This could happen if you are reading an archived version of the page. If it the exchange rates are older than a day, I don't show them.

```
fetch('/api/exchangerates.json')
      .then(...)
      .then(data => {
        const dataAgeInHours = ((new Date(data.timestamp * 1000)).get
        if(dataAgeInHours >= 24){
          return;
        }
        ...
      });
```

I also check if the API response is valid, since fetch does not fail on invalid responses.

```
fetch('/api/exchangerates.json')
  .then(response => {
    if(!response.ok){
      throw new Error('Cannot retrieve exchange rates.');
    }
    .then(...);
```

If the amounts are larger than 100 units of a currency, I don't show cents. I only show the significant digits, to avoid false precision.

```
const asCurrency = (usdValue, currencyCode) => {
  const value = usdValue * data.rates[currencyCode];
  const showCents = value < 100;
  return Intl.NumberFormat('en-US', {style: 'currency', currency: cur
};
```

Bringing it all together

This is the code that made it to production:

```
fetch('/api/exchangerates.json')
      .then(response => {
        if(!response.ok){
          throw new Error('Cannot retrieve exchange rates.');
        }
        return response.json()
      })
      .then(data => {
        const dataAgeInHours = ((new Date(data.timestamp * 1000)).get
        if(dataAgeInHours >= 24){
          return;
        }
        const defaultCurrencyCodes = ["USD", "GBP", "INR"];
        const countryCodeToCurrencyCode = {"AF":"AFN","AL":"ALL","DZ"
        const selectedCurrencyCodes = new Set(
          navigator.languages.map(l => countryCodeToCurrencyCode[l.su
            .filter(Boolean)
            .concat(defaultCurrencyCodes)
        );
        const asCurrency = (usdValue, currencyCode) => {
          const value = usdValue * data.rates[currencyCode];
          const showCents = value < 100;
          return Intl.NumberFormat('en-US', {style: 'currency', curre
        };
```

```
    const usdValue = Number(el.textContent.replace(',', '')) / 
    el.dataset.currencies = Array.from(selectedCurrencyCodes).s
  });
});
```

How to tighten the steering head bearing on a V-Strom →