Brief History of Moments

How Not to Handle Time in Javascript

Time seems to have elluded humans for a *long while* now.

- See physicists who claim time doesn't exist
- and those who claim it does

Of course we get it wrong.

Dealing with *time* in code is a difficult thing.

- See year 2k bug
- See year 2038 problem

Even more so as the system you are interacting with or designing is complex

As a beginner developer, I did my fair share of mistakes

One of them costed my employer £500

```
const currentDate = `${res["year"]}-0${res["mon"]}-${res["mday"]}`

if (getUnixTimestamp === currentDate)
    // 'The player played this game today'
else
    loseEmployerMoney()
```

Just in the first day

One weekend later, ramped up to £5,000.

Showing **no** signs of *slowing* down.

I was ashamed, angry, lost, frustrated

Cognitive effort to deal even with simle time calculations is high

Do not make it even harder by doing things like

```
const tzoffset = (new Date()).getTimezoneOffset() * 60000 // offset in milliseconds
const localISOTime = (new Date(Date.now() - tzoffset)).toISOString().slice(0, -1)
const currentDate = localISOTime.match(/\d{4}(-\d\d){2}/g)[0]
const midnightExpiry = d.setUTCHours(23,0,0,0);
const datePlayed = timeMethods.unixToDate(midnightExpiry - 1)
console.log(datePlayed); // !!!
```

Especially *misterious* midnightExpiry variable name must be important

Never ever..

- do your own calculations
- compare 2 "dates" when actually comparing "strings"

But Do..

- rely on external helpers (moment.js)
- external APIs that count the time for your company / group [^]

[^] this might be specific if worldwide, timezone restricted etc