Open Opened 3 hours ago by **Rubén Montero**

Formatear una fecha

Resumen

- Añadiremos una `prop` a nuestro componente `TimestampChecker` para que, si es `true`, formatee la fecha
- Usaremos de nuevo el `TimestampChecker` en otro `<div>`, pasando dicha `prop` (`formatDate={true}`)

Descripción

El formateo de fechas es un problema recurrente en programación. Mientras nosotros acostumbramos a escribir las fechas comenzando por el día, el mes y el año:

> 7/10/2023

...en Estados Unidos, primero se escribe el mes y luego el día:

> 10/7/2023

Es sólo uno de los múltiples problemas que nos podemos encontrar.

En JavaScript, al igual que en muchos otros lenguajes, *existe una clase destinada a representar fechas y horas*. Se trata de `Date`. Podemos construir un nuevo `Date` así:

```
const date = new Date()
```

Y con ello, tenemos un objeto que *representa la fecha actual*. Puedes consultar diferentes formas de formatear un Date, aunque nosotros nos quedaremos únicamente con `.toLocaleString()`. Un ejemplo de uso sería este:

```
const date = new Date()

// La siguiente línea imprime la fecha actual así: 7/2/2021, 14:05:07
console.log(date.toLocaleString())
```

¿Se puede construir un objeto `Date` de otra forma?

Sí. Puedes construir un objeto `Date` a partir de un *timestamp*. Basta con pasarlo al método constructor. Por ejemplo:

```
const date = new Date(1654600788000); // Devuelve la fecha representada por ese timestamp
```

¿El *timestamp* de http://time.akamai.com vale?

Sí.

Y no.

Es un *timestamp* válido, pero está representado en *segundos*, mientras que el objeto `Date` espera un *timestamp* en *milisegundos*. Fácil. Sólo hay que multiplicarlo por mil.

La tarea

Modifica `TimestampChecker.js` para que, si viene `props.formatDate`, **formatee** la fecha como se indica arriba. **Así**:

```
import axios from "axios";
import { useState, useEffect } from "react"

const TimestampChecker = (props) => {
    const [time, setTime] = useState(undefined);

    useEffect(() => {
        // Esto se ejecuta cuando sucede el efecto. ¡Enviamos la petición!
        axios.get("http://time.akamai.com").then(response => {
            // Esto se ejecuta cuando la respuesta HTTP (será response.data) nos llega
```

```
-           setTime(response.data);
+           if (props.formatDate === true) {
+               const timestampInSeconds = parseInt(response.data);
+               const timestampInMillis = timestampInSeconds * 1000;
+               const date = new Date(timestampInMillis);
+               setTime(date.toLocaleString());
+           } else {
+               setTime(response.data);
+           }
        })
    }, [])

    return <div data-cy='timestampChecker'>
        <p data-cy='title'>El timestamp según Akamai:</p>
        <p data-cy='timestamp'>{time}</p>
      </div>
}

export default TimestampChecker;
```

Para finalizar, en `Examples.js`, **añade** un nuevo `<div>` *a continuación* (debajo) del correspondiente al ejercicio anterior. En él, **incluye** el componente `<TimestampChecker />`, esta vez, pasando la *prop* `formatDate={true}` para verificar el nuevo funcionamiento, y además un encabezado. **Así**:

```
  <div data-cy='issue15div'>
    <h1>Ejercicio 15</h1>
    <TimestampChecker formatDate={true} />
  </div>
```

Verifica su funcionamiento en http://localhost:3000/examples.

🏆 Por último

Sube tus cambios al repositorio en un nuevo *commit*.

Rubén Montero @ruben.montero changed milestone to %Sprint 3 3 hours ago