Exercicio 61: Tratar as datas *timestamp* en JavaScript

Obxectivo:

Imos a usar os **datos da API da aplicación do tempo** do ***Exercicio 60*** para practicar cos **métodos de tratamento do tempo *timestamp*** en JavaScript. As tarefas que temos que realizar son as seguintes:

- Engadir a **data do día completa**: día da semana, día do mes, mes do ano e ano en formato completo debaixo do nome do lugar. *Exemplo: Luns, 31 de abril de 2025.*
- Engadir a **hora do abrente** proporcionada pola API do tempo lexible por humanos.
- Engadir a **hora do solpor** proporcionada pola API do tempo lexible por humanos.
- Crear un **fondo dinámico** para que cambie se é de día ou de noite.

Descrición:

En JavaScript, as *timestamps* representan o número de milisegundos transcorridos dende o **1 de xaneiro de 1970** (época Unix). Para traballar con eles, utilizamos o obxecto Date, que ofrece métodos útiles para manipular e obter información sobre datas.

Métodos principais para tratar *timestamps* en JavaScript

1. **Crear un obxecto Date:**
    - Sen argumentos: devolve a data e hora actual.

```
let agora = new Date();

console.log(agora); // Data e hora actual
```

    - A partir dun *timestamp*:

```
let dataDesdeTimestamp = new Date(1672531200000); // Milisegundos

console.log(dataDesdeTimestamp); // Data correspondente
```

2. **Obter o *timestamp* actual:**
    - Co método **Date.now():**

```
let timestampActual = Date.now();

console.log(timestampActual); // Milisegundos desde 1970
```

3. **Obter datas a partires de *timestamp***:

    - **Ano:**

    getFullYear(): Devolve o ano completo (por exemplo, 2025).

```
const date = new Date();
console.log(date.getFullYear()); // 2025
```

    - **Mes**:

    getMonth(): Devolve o mes como un número entre 0 (xaneiro) e 11 (decembro).

```
const date = new Date();
console.log(date.getMonth()); // 2 (marzo, xa que os meses comezan en 0)
```

    - **Día do mes**:

    getDate(): Devolve o día do mes (1 a 31).

```
const date = new Date();
console.log(date.getDate()); // 30
```

    - **Día da semana**:

    getDay(): Devolve o día da semana como un número entre 0 (domingo) e 6 (sábado).

```
const date = new Date();
console.log(date.getDay()); // 0 (domingo)
```

    - **Hora**:

    getHours(): Devolve a hora actual en formato de 24 horas (0 a 23).

```
const date = new Date();
console.log(date.getHours()); // 12
```

- **Minutos e segundos**:

getMinutes(): Devolve os minutos actuais.

getSeconds(): Devolve os segundos actuais.

```
const date = new Date();
console.log(date.getMinutes()); // Exemplo: 56
console.log(date.getSeconds()); // Exemplo: 30
```

- **Milisegundos:**

getMilliseconds(): Devolve os milisegundos actuais.

```
const date = new Date();
console.log(date.getMilliseconds()); // Exemplo: 999
```

Consellos adicionais

- Os meses en JavaScript comezan en 0 (xaneiro = 0, decembro = 11).
- Os días da semana tamén se contan desde 0 (domingo = 0, sábado = 6).

Con estes métodos, podes manipular datas e tempos de xeito eficiente sen depender de librarías externas.

Formatear datas lexibles por humanos:

Para mostrar as datas con formato completo en español, podes usar o **método *toLocaleDateString()*** con opcións específicas.

Entrega: Unha vez completado todo **subiremos a carpeta *Exercicio61*** ao espazo para entrega de exercicios que temos no OneDrive de Microsoft Teams.

Ejercicio 61: Tratar las fechas *timestamp* en JavaScript

Objetivo:

Utilizar los datos de la API de la aplicación meteorológica del Ejercicio 60 para practicar con los métodos de manipulación de fechas *timestamp* en JavaScript.

Tareas:

- **Mostrar la fecha completa** en formato: día de la semana, día del mes, mes y año. Ejemplo: *Lunes, 31 de abril de 2025*.
- Convertir la **hora del amanecer** proporcionada por la API a un formato legible por humanos.
- Convertir la **hora del atardecer** proporcionada por la API a un formato legible por humanos.
- Elaborar un **fondo dinámico** que cambie en función de si es día o noche.

Descripción:
En JavaScript, las marcas de tiempo (*timestamps*) representan el número de milisegundos transcurridos desde el 1 de enero de 1970 (época Unix). Para trabajar con ellas, se utiliza el objeto Date, que incluye métodos útiles para manipular y obtener información sobre fechas.

Métodos principales para trabajar con *timestamps* en JavaScript

1. **Crear un objeto Date:**
   - Sin argumentos: devuelve la fecha y hora actual.

```
let ahora = new Date();
console.log(ahora); // Fecha y hora actual
```

   - A partir de un *timestamp*:

```
let fechaDesdeTimestamp = new Date(1672531200000); // Milisegundos
console.log(fechaDesdeTimestamp); // Fecha correspondiente
```

2. **Obtener el *timestamp* actual:**
   - Con Date.now():

```
let timestampActual = Date.now();
console.log(timestampActual); // Milisegundos desde 1970
```

3. **Obtener información específica de una fecha:**

   - **Año completo**:

     getFullYear(): Devuelve el año completo (por ejemplo, 2025).

```
const date = new Date();
console.log(date.getFullYear()); // Ejemplo: 2025
```

   - **Mes** (0 = enero, 11 = diciembre):

     getMonth(): Devuelve el mes como un número entre 0 (enero) y 11 (diciembre).

```
console.log(date.getMonth()); // Ejemplo: 2 (marzo)
```

   - **Día del mes**:

     getDate(): Devuelve el día del mes (1 a 31).

```
console.log(date.getDate()); // Ejemplo: 30
```

   - **Día de la semana** (0 = domingo, 6 = sábado):

     getDay(): Devuelve el día de la semana como un número entre 0 (domingo) y 6 (sábado).

```
console.log(date.getDay()); // Ejemplo: 0 (domingo)
```

   - **Hora**:

     getHours(): Devolve a hora actual en formato de 24 horas (0 a 23).

```
console.log(date.getHours()); // Ejemplo: 12
```

- **Minutos y segundos**:

getMinutes(): Devolve os minutos actuais.

getSeconds(): Devolve os segundos actuais.

```
console.log(date.getMinutes()); // Ejemplo: 56
console.log(date.getSeconds()); // Ejemplo: 30
```

- **Milisegundos**:

getMilliseconds(): Devolve os milisegundos actuais.

```
console.log(date.getMilliseconds()); // Ejemplo: 999
```

Consejos adicionales:

- Los meses en JavaScript comienzan en 0 (enero) y terminan en 11 (diciembre).
- Los días de la semana también se cuentan desde 0 (domingo) hasta 6 (sábado).

Formatear fechas legibles por humanos:

Para mostrar las fechas en español con formato completo, se puede usar el método ***toLocaleDateString()*** con opciones específicas.

Entrega:

Subir la carpeta "***Ejercicio61***" al espacio de entrega en **OneDrive** dentro de **Microsoft Teams** una vez completado.