09

Mãos na Massa: Enviando dados para o processo principal

> Começando deste ponto? Você pode fazer o DOWNLOAD (https://s3.amazonaws.com/caelum-online-public/electron/cap03/stages/alura-timer-fim-cap03.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

Neste capítulo queremos salvar os dados dos cursos estudos em nosso timer. Para isto, vamos criar arquivos JSON com o nome do curso e o tempo estudado em cada um deles toda vez que pausarmos um curso.

Sabemos que o momento que pausamos o curso, a função executada é a `timer.parar()`, então vamos modifica-la para enviar um evento para o processo principal, informando que queremos salvar o tempo e o curso estudado.

1- Em seu `timer.js` importe o módulo `ipcRenderer` para fazermos a comunicação com o processo principal:

```
//timer.js
const { ipcRenderer } = require('electron')
```

2- Na função `parar()`, vamos enviar para o módulo principal o evento de `curso-parado`:

```
//timer.js
...
    },parar(){
        clearInterval(timer);
        ipcRenderer.send('curso-parado');
    },
...
```

3- Como temos que enviar o nome do curso e o tempo estudado, vamos obter estas duas informações. Primeiro, o nome do curso, que vamos extrair do HTML no `renderer.js`, e passar para a função `timer.parar()` quando chamamos-a:

```
//renderer.js
let curso = document.querySelector('.curso');
```

```
//renderer.js
...
    if(play){
        timer.parar(curso.textContent);
        play = false;
    }
...
```

4- Alterando a função `timer.parar()` para que ela receba o nome do curso como parâmetro e envie junto do evento:

```
//timer.js
...
```

```
    },parar(curso){
        clearInterval(timer);
        ipcRenderer.send('curso-parado', curso);
    },
...
```

5- Agora para obter o tempo de estudo no curso, vamos criar a variável `tempoEstudado` e utilizar nossa função `segundosParaTempo` para converter o tempo:

```
//timer.js
...
    },parar(curso){
        clearInterval(timer);
        let tempoEstudado = this.segundosParaTempo(segundos);
        ipcRenderer.send('curso-parado', curso, tempoEstudado);
    },
...
```

6- Por último, vamos para o processo principal escutar este evento e exibir os dados que recebemos:

```
//main.js
ipcMain.on('curso-parado', (event, curso, tempoEstudado) => {
    console.log(`O curso ${curso} foi estudado por ${tempoEstudado}`);
})
```