03

Ciclo de vida das janelas

Transcrição

Vamos resolver o problemas das múltiplas janelas de **Sobre**. Isso está acontecendo por que toda vez que clicamos no link **Sobre**, o seguinte código é executado:

```
// main.js

ipcMain.on('abrir-janela-sobre', () => {
    let sobreWindow = new BrowserWindow({
        width: 300,
        height: 220
    });
    sobreWindow.loadURL(`file://${__dirname}/app/sobre.html`);
});
```

Esse é o código que escuta o evento `abrir-janela-sobre`, e instancia um `BrowserWindow`, criando uma nova janela. Logo, toda vez que o evento `` `abrir-janela-sobre `` é enviado, uma nova janela é criada.

Ao invés de criarmos uma janela toda vez que o evento for enviado, vamos criá-la somente uma vez, e se ela já estiver criada, pulamos o código que instancia o `BrowserWindow`.

Lidando com múltiplas janelas

O que podemos fazer é mover a inicialização da variável **`sobreWindow`** para **fora** do evento escutado, com o valor **`null`**:

```
// main.js

let sobreWindow = null;

ipcMain.on('abrir-janela-sobre', () => {
    sobreWindow = new BrowserWindow({
        width: 300,
        height: 220
    });
    sobreWindow.loadURL(`file://${__dirname}/app/sobre.html`);
});
```

E dentro da função, fazemos o seguinte teste: se **`sobreWindow`** for **`null`**, nós criamos uma nova janela:

```
// main.js

let sobreWindow = null;

ipcMain.on('abrir-janela-sobre', () => {
    if(sobreWindow == null) {
        sobreWindow = new BrowserWindow({
            width: 300,
```

```
            height: 220
        });
    }
    sobreWindow.loadURL(`file://${__dirname}/app/sobre.html`);
});
```

Ou seja, na primeira vez que clicarmos no link **Sobre**, `sobreWindow` será nulo, logo a janela será criada. A partir da segunda vez, a variável não será mais nula, logo a janela não será criada, somente carregará a página.

Ao testar a aplicação, clicamos no link **Sobre** e a janela é aberta. Mas ao fechar a janela, e clicar novamente no link, recebemos um erro, dizendo que o objeto foi destruído! Como assim?

Quando clicamos no **X** para fechar a janela, o objeto será destruído. Logo, quando o Electron tentar criar um novo `BrowserWindow`, ele não irá conseguir, pois a variável `sobreWindow` não existe mais.

Evitando que a variável seja destruída

Para resolver esse problema, precisamos evitar que a variável seja destruída após o fechamento da janela. Para isso, podemos escutar um evento seu, o evento de `closed`. Ou seja, quando a janela for fechada, nós fazemos algo:

```
// main.js

let sobreWindow = null;

ipcMain.on('abrir-janela-sobre', () => {
    if(sobreWindow == null) {
        sobreWindow = new BrowserWindow({
            width: 300,
            height: 220
        });
        sobreWindow.on('closed', () => {

        });
    }
    sobreWindow.loadURL(`file://${__dirname}/app/sobre.html`);
});
```

Quando a janela for fechada, ao invés de eliminar a variável, vamos colocá-la como nula novamente! Assim evitamos que o JavaScript destrua a nossa variável:

```
// main.js

let sobreWindow = null;

ipcMain.on('abrir-janela-sobre', () => {
    if(sobreWindow == null) {
        sobreWindow = new BrowserWindow({
            width: 300,
            height: 220
        });
        sobreWindow.on('closed', () => {
            sobreWindow = null;
```

```
        });
    }
    sobreWindow.loadURL(`file://${__dirname}/app/sobre.html`);
});
```

Ao testar nossa aplicação, vemos que agora está tudo funcionando corretamente! Não conseguimos abrir múltiplas janelas e nem acontece um erro quando fechamos e abrimos novamente a janela **Sobre**.