05

Propriedades da Browser Window

Transcrição

Resolvido o problema das múltiplas janelas, outra característica atual da nossa aplicação é que, com a janela de Sobre aberta, caso cliquemos fora dela (mas ainda no aplicativo), ela vai para trás da janela principal, ficando escondida.

Se é uma janela que gostaríamos que o usuário fique lendo, seria bom que ela ficasse sempre em primeiro plano, na frente da janela principal da aplicação.

Janela sempre em primeiro plano

Para implementar essa funcionalidade, podemos configurar a janela, já que essas janelas são extremamente configuráveis. E uma dessas configurações, podemos passar dentro do objeto `BrowserWindow`, na hora de instanciá-lo. A propriedade que estamos querendo é a `alwaysOnTop`, que faz com que a janela sempre fique em primeiro plano. Logo, vamos habilitá-la, passando o valor `true` para a propriedade:

```
// main.js

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

Agora fazemos o mesmo teste que anteriormente, vemos que a janela de Sobre continua por cima da janela principal, logo, resolvemos a questão. Mas ao clicar fora da aplicação, a janela continua por cima, seja do navegador, do editor de texto, de qualquer programa que esteja aberto. Essa é uma característica dessa propriedade `alwaysOnTop`.

Além disso, há diversas outras configuração que podemos fazer na `BrowserWindow`, se a janela vai recarregar ou não, se ela vai poder mudar de tamanho, se vai começar aparecendo no meio da janela ou no canto, entre diversas outras configurações. Podemos vê-las na documentação do Electron (https://electron.atom.io/docs/), mais precisamente na seção ***BrowserWindow*** (https://electron.atom.io/docs/api/browser-window/), onde podemos ver as **configurações** (https://electron.atom.io/docs/api/browser-window/#new-browserwindowoptions) que podemos fazer.

Fechando uma janela

Uma outra configuração que podemos fazer, é remover a barra superior da janela, que contém os botões de maximizar, minimizar e fechar. Fazemos isso com a propriedade `frame`, passando o valor `false` para ela;

```
// main.js

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

Só que sem o botão de fechar a janela, como fazemos para fechá-la? Podemos criar um botão para fechar a janela, lá na página **sobre.html**:

```
<!-- sobre.html -->

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Sobre o Alura Timer</title>
</head>
<body>
    <p>O Alura timer é para você guardar seus tempos de estudo na Alura.</p>
    <a href="#" id="link-fechar">Fechar</a>
</body>
</html>
```

Se agora temos um link, temos que detectar um clique nele. Logo, dentro de **app/js/**, vamos criar o **sobre.js**. Antes de implementar o código JavaScript, vamos importar o arquivo na página:

```
<!-- sobre.html -->

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Sobre o Alura Timer</title>
</head>
<body>
    <p>O Alura timer é para você guardar seus tempos de estudo na Alura.</p>
    <a href="#" id="link-fechar">Fechar</a>
    <script src="js/sobre.js"></script>
</body>
</html>
```

Já no **sobre.js**, selecionamos o link e escutamos o seu evento de `click`:

```
// sobre.js

let linkFechar = document.querySelector("#link-fechar");

linkFechar.addEventListener('click', function () {

})
```

Vimos que o **main.js** é o responsável pelo ciclo das janelas, logo ele que é capaz de fechar as janelas. Para abrir uma janela, nós enviamos um evento para ele, e para fechar não será diferente, importamos o `ipcRenderer` e utilizamos-o para enviar o evento `fechar-janela-sobre` para o processo principal:

```
// sobre.js

const { ipcRenderer } = require('electron');

let linkFechar = document.querySelector("#link-fechar");

linkFechar.addEventListener('click', function () {
    ipcRenderer.send('fechar-janela-sobre');
})
```

E no processo principal, o **main.js**, escutamos o evento `fechar-janela-sobre`. Para fechar a janela, nós chamamos o seu método `close()`:

```
// main.js

ipcMain.on('fechar-janela-sobre', () => {
    sobreWindow.close();
});
```

Ao testar, conseguimos fechar a janela de **Sobre**. Então, é importante dominarmos esse tipo de comunicação entre os processos, pois há coisas que só conseguimos fazer no processo principal, e outras que só conseguimos fazer no processo de *renderer*.