

Modificando a Traybar

Transcrição

Sabemos que o menu é montado no processo principal, então, para adicionar o curso ao menu, assim que ele for adicionado ao jogo, devemos enviar um evento para o processo principal, que por sua vez será capaz de alterar o menu.

Então, no **renderer.js**, enviamos o evento `curso-adicionado`, juntamente com o nome do novo curso, para o processo principal:

```
// renderer.js
// restante do código omitido

botaoAdicionar.addEventListener('click', function() {
  let novoCurso = campoAdicionar.value;
  curso.textContent = novoCurso;
  tempo.textContent = '00:00:00';
  campoAdicionar.value = '';
  // enviando o evento para o processo principal
  ipcRenderer.send('curso-adicionado', novoCurso);
});
```

Agora, no processo principal (**main.js**), nós escutamos esse evento:

```
// main.js
// restante do código omitido

ipcMain.on('curso-adicionado', (event, novoCurso) => {

});
```

Agora, nós devemos adicionar o curso ao menu. O responsável por isso é o **template.js**, é a partir dele nós montamos o *tray template*, que utilizamos para montar o menu. Ou seja, quando dizemos que queremos adicionar um novo curso ao menu, devemos adicioná-lo ao **template**.

Para isso, no **template.js**, vamos criar a função `adicionaCursoNoTray`, que receberá por parâmetro o curso a ser adicionado:

```
// template.js

const data = require('./data');

module.exports = {
  geraTrayTemplate(win) {
    // código da função omitido
  },
  adicionaCursoNoTray(curso) {

  }
}
```

E no **main.js**, pedimos para o **template.js** adicionar o curso e geramos novamente o menu:

```
// main.js
// restante do código omitido

ipcMain.on('curso-adicionado', (event, novoCurso) => {
  let novoTemplate = templateGenerator.adicionaCursoNoTray(novoCurso);
  let novoTrayMenu = Menu.buildFromTemplate(novoTemplate);
  tray.setContextMenu(novoTrayMenu);
});
```

Resta agora implementar a função `adicionaCursoNoTray`.

Adicionando um novo curso ao Tray Menu

Na função `adicionaCursoNoTray`, devemos alterar o array `template`, mas que é criado dentro da função `geraTrayTemplate`. Para conseguirmos manipular esse array em outra função, vamos criar uma variável fora das funções, e antes de retornar o `template` na função `geraTrayTemplate`, nós vamos atribuí-lo à nova variável:

```
// template.js

const data = require('./data');

module.exports = {
  templateInicial: null,

  geraTrayTemplate(win) {
    // restando do código da função omitido
    this.templateInicial = template;
    return template;
  },
  adicionaCursoNoTray(curso) {

  }
}
```

A partir do momento que temos acesso ao `templateInicial`, basta adicionar nele um objeto com as características do novo curso e retorná-lo ao final:

```
// template.js

const data = require('./data');

module.exports = {
  templateInicial: null,

  geraTrayTemplate(win) {
    // código da função omitido
  },
  adicionaCursoNoTray(curso) {
```

```

    this.templateInicial.push({
      label: curso,
      type: 'radio',
      click: () => {
        win.send('curso-trocado', curso);
      }
    });

    return this.templateInicial;
  }
}

```

Assim que o curso for adicionado ao menu, queremos que ele fique marcado como *checked*, para isso vamos adicionar a propriedade de mesmo nome a ele. E como precisamos ter acesso à janela, vamos recebê-la por parâmetro:

```

// template.js

const data = require('./data');

module.exports = {
  templateInicial: null,

  geraTrayTemplate(win) {
    // código da função omitido
  },
  adicionaCursoNoTray(curso, win) {
    this.templateInicial.push({
      label: curso,
      type: 'radio',
      checked: true,
      click: () => {
        win.send('curso-trocado', curso);
      }
    });

    return this.templateInicial;
  }
}

```

Por fim, no **main.js**, para podermos enviar a `mainWindow` para a função `adicionaCursoNoTray`, vamos declará-la fora de qualquer função, e dentro do *on ready*, nós somente atribuímos o valor a ela

```

// main.js

// restante do código omitido

// declarando a mainWindow
let mainWindow = null;
app.on('ready', () => {
  console.log('Aplicação iniciada');

  // atribuindo o valor, e não mais declarando
  mainWindow = new BrowserWindow({
    width: 600,

```

```
        height: 400,
      });
      // restante do código omitido
    });

    // restante do código omitido

    ipcMain.on('curso-adicionado', (event, novoCurso) => {
      // enviando a mainWindow para a função adicionaCursoNoTray
      let novoTemplate = templateGenerator.adicionaCursoNoTray(novoCurso, mainWindow);
      let novoTrayMenu = Menu.buildFromTemplate(novoTemplate);
      tray.setContextMenu(novoTrayMenu);
    });
```

Devemos sempre ter o cuidado de declarar globalmente as variáveis que representam atributos globais no Electron, como menus e janelas.

Ao executar a aplicação, ao adicionar o curso, vemos que ele aparece selecionado no menu. Além disso, conseguimos trocar de cursos, inclusive voltar ao curso recém-adicionado, com tudo funcionando corretamente.