09

Sobre interface de funções

Walter decidiu praticar um pouco sobre interface de funções e criou o seguinte código:

```
function executaAssincrono(cb: Function) {

    setTimeout(() => cb('terminou'), 0);
}

let callback1 = ((resultado: any) => alert(resultado));
let callback2 = ((resultado: any) => alert(`**${resultado}**`));

executaAssincrono(callback1);
executaAssincrono(callback2);
```

O problema é que ele não sabe como criar agora a inteface `MeuCallback` que substitua o `Function` adotado por `executaAssincrono` .

Marque a opção que cria e utiliza corretamente a interface:

Selecione uma alternativa

A

```
interface MeuCallback {

    (mensagem: string): void;
}


function executaAssincrono(cb: MeuCallback) {

    setTimeout(() => cb('terminou'), 0);
}

let callback1: MeuCallback = resultado => alert(resultado);
let callback2: MeuCallback = resultado => alert(`**${resultado}**`);

executaAssincrono(callback1);
executaAssincrono(callback2);
```

B

```
interface MeuCallback {

    func(mensagem: string): void;
}

function executaAssincrono(cb: MeuCallback) {

    setTimeout(() => cb('terminou'), 0);
}

let callback1: MeuCallback = resultado => alert(resultado);
let callback2: MeuCallback = resultado => alert(`**${resultado}**`);
```

```
executaAssincrono(callback1);
executaAssincrono(callback2);
```

C

```
interface MeuCallback {

    (mensagem: string): void;
}


function executaAssincrono(cb: MeuCallback) {

    setTimeout(() => cb('terminou'), 0);
}

let callback1 = resultado => alert(resultado);
let callback2 = resultado => alert(`**${resultado}**`);

executaAssincrono(callback1);
executaAssincrono(callback2);
```