07

Exportando funcionalidades de um módulo (ES5, ainda!)

Valentina criou duas funções para lidar com conversões de um número em real para número e vice-versa:

```
// conv.js

let simboloMoeda = 'R$ ';

function numeroParaReal(numero) {

    return simboloMoeda + numero.toFixed(2).replace('.', ',');
}

function realParaNumero(texto) {

    return texto.replace(simboloMoeda, '').replace(',', '.');
}
```

Podemos até testar através do console do Chrome:

```
let real = 'R$ 100,20';
let realConvertidoEmNumero = realParaNumero(real);
alert(realConvertidoEmNumero); // exibe 100.20

let numero = 200.15;
let numeroConvertidoEmReal = numeroParaReal(numero);
alert(numeroConvertidoEmReal); // exibe R$ 200,15
```

É claro que existem várias maneiras de se resolver esse problema, essa é uma das soluções. Contudo, veja que temos a variável `simboloMoeda` e as duas funções no escopo global. Sendo assim, podemos alterar o símbolo da moeda a qualquer momento:

```
var simboloMoeda = '$';
```

Veja que agora o símbolo da moeda é dólar! Com certeza não queremos que essa mudança seja possível em nosso código. Sabemos que se declararmos variáveis (se você usar `let` ou `var`) e funções dentro de outra função, elas pertencerão ao escopo da função e **não** pertencerão ao escopo global. Inclusive aprendemos a utilizar uma **IIFE**, uma função anônima que se invoca automaticamente para criar esse escopo:

```
// conv.js

(function() {

    let simboloMoeda = 'R$ ';

    function numeroParaReal(numero) {

        return simboloMoeda + numero.toFixed(2).replace('.', ',');
```

```
    }

    function realParaNumero(texto) {

        return texto.replace(simboloMoeda, '').replace(',', '.');
    }
})();
```

Veja que agora, quando nosso script for carregado, a IIFE será invocada e tudo que estiver dentro dela não será mais acessível globalmente. Desse jeito, ninguém poderá mais alterar `simboloMoeda`, **mas também não conseguirá chamar as funções** `numeroParaReal` e `realParaNumero`. Resolvemos um problema e criamos outro.

Você aprendeu neste capítulo sobre o padrão de projeto ***Module Pattern***. O que fizemos até agora foi criar um módulo, uma unidade de código confinada, **mas que não exporta qualquer funcionalidade**.

Qual das opções abaixo altera corretamente nosso módulo para que exporte as funcionalidades `numeroParaReal` e `realParaNumero`, mas mantendo `simboloMoeda` inacessível através do escopo global?

Selecione uma alternativa

A

```
(function() {

    let simboloMoeda = 'R$ ';

    return function numeroParaReal(numero) {

        return simboloMoeda + numero.toFixed(2).replace('.', ',');
    }

     function realParaNumero(texto) {

        return texto.replace(simboloMoeda, '').replace(',', '.');
    }
})();

// exemplo de uso

let real = 'R$ 100,20';
let realConvertidoEmNumero =
     formatadorDeMoedas.realParaNumero(real);

alert(realConvertidoEmNumero);

let numero = 200.15;
let numeroConvertidoEmReal =
    formatadorDeMoedas.numeroParaReal(numero);

alert(numeroConvertidoEmReal);
```

B

```
var formatadorDeMoedas = (function() {

    let simboloMoeda = 'R$ ';
    let modulo = {};
```

```
    modulo.numeroParaReal = numero => {

        return simboloMoeda + numero.toFixed(2).replace('.', ',');
    }

    modulo.realParaNumero = texto => {

        return texto.replace(simboloMoeda, '').replace(',', '.');
    }

    return modulo;
})();

// exemplo de uso

let real = 'R$ 100,20';
let realConvertidoEmNumero =
     formatadorDeMoedas.realParaNumero(real);

alert(realConvertidoEmNumero);

let numero = 200.15;
let numeroConvertidoEmReal =
    formatadorDeMoedas.numeroParaReal(numero);

alert(numeroConvertidoEmReal);
```

C

```
(function() {

    let simboloMoeda = 'R$ ';

    return function numeroParaReal(numero) {

        return simboloMoeda + numero.toFixed(2).replace('.', ',');
    }

    return function realParaNumero(texto) {

        return texto.replace(simboloMoeda, '').replace(',', '.');
    }
})();

// exemplo de uso

let real = 'R$ 100,20';
let realConvertidoEmNumero =
     formatadorDeMoedas.realParaNumero(real);

alert(realConvertidoEmNumero);

let numero = 200.15;
let numeroConvertidoEmReal =
    formatadorDeMoedas.numeroParaReal(numero);

alert(numeroConvertidoEmReal);
```