12

# Criando o "ajudante"

Que tal um exemplo diferente do que foi apresentado no vídeo? Caso você queira implementá-lo, sugiro que você crie outro projeto para não interferir no código do projeto deste curso, combinado?

Temos a página `upload.html`, na qual o usuário insere os dados do arquivo que deseja fazer upload. É claro que é um formulário de mentirinha, até porque o processo de upload é mais complexo, mas a ideia aqui é exercitar seu conhecimento adquirido em JavaScript:

```
<!-- upload.html -->
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Upload de arquivo</title>
</head>
<body>
    <label>Informações do arquivo</label>
    <input class="dados-arquivo" placeholder="formato: nome/tamanho/tipo">

    <button onclick="arquivoController.envia()"  >Enviar</button>

    <script src="Arquivo.js"></script>
    <script src="ArquivoController.js"></script>
    <script>
        let arquivoController = new ArquivoController();
    </script>
</body>
</html>
```

Aprendemos a organizar nosso código no padrão MVC e por isso temos uma classe que é uma abstração de um arquivo. Suas propriedade são **nome**, **tamanho**, **tipo**:

```
class Arquivo {

    constructor(nome, tamanho, tipo) {
        this._nome = nome;
        this._tamanho = tamanho;
        this._tipo = tipo;
    }

    get nome() {
        return this._nome;
    }

    get tamanho() {
        return this._tamanho;
    }

    get tipo() {
        return this._tipo;
```

```
    }
}
```

É claro, temos um controller que sabe instanciar um modelo da classe `Arquivo` com os dados do formulário, toda vez que o usuário clicar no botão "Enviar":

```
class ArquivoController {

    constructor() {
        this._inputDados = document.querySelector('.dados-arquivo');
    }

    envia() {
        //cria um Arquivo com as suas propriedades;
        this._limpaFormulario();
        // exibe mensagem no console com os dados do arquivo.
    }

    _limpaFormulario() {
        this._inputDados.value = '';
        this._inputDados.focus();
    }
}
```

Veja que o método `envia`, de `ArquivoController`, não está completo porque temos um problema. O **problema** é que o designer achou melhor ter **apenas um campo** onde o usuário digita na sequência o **nome**, o **tamanho** e o **tipo** do arquivo. Infelizmente o construtor de `Arquivo` não está preparado para receber a string com todos os dados, mas cada um em separado como parâmetro do construtor. E por fim, para complicar só mais um pouquinho, a entrada do usuário deve ser toda considerada em caixa alta, ou seja, maiúsculo.

Sua tarefa é implementar o método `envia` de `ArquivoController`. Ele deverá ler a entrada do usuário e adequá-la ao construtor de `Arquivo`. Assim que você conseguir criar uma instância de arquivo, você deve imprimir seus dados no console.

Lembre-se que o usuário digita no campo de entrada os dados do arquivo no formato: *nome* / *tamanho* / *tipo* e deve estar em caixa alta!