

## Colocando erro no HTML

### Transcrição

Vamos colocar a mensagem de erro no HTML, para a qual teremos acesso criando a variável `sugestao`. Inicialmente, seria interessante que esta mensagem viesse escondida via CSS por padrão. Sendo assim, iremos a `index.html` para alterarmos `contatoCampo--msg`.

Usaremos o atalho "Ctrl + E" para acessarmos a classe, e trocaremos `inline-block` em `display` por `none`. Com isto, ela ficará escondida inclusive para o leitor de telas. Voltaremos a `mailcheck-config.js` e, a partir de agora, se houver alguma sugestão, queremos que ela apareça. Indicaremos isto da seguinte maneira:

```
document.querySelector('#email').addEventListener('blur', function() {
  Mailcheck.run({
    email: campoEmail.value,
    domains: domains, // optional
    topLevelDomains: topLevelDomains, // optional
    secondLevelDomains: secondLevelDomains, // optional
    distanceFunction: superStringDistance, // optional
    suggested: function(suggestion) {
      sugestao.style.display = 'inline-block';
    }
  });
});
```

Vamos testar? Salvaremos, retornaremos a `index.html`, incluímos algo aleatório dentre as `tags` `<span>` ("asdasd"), salvaremos e abriremos o navegador para testarmos incluindo o email "meunomelegal@gmail.com", ao que receberemos o texto colocado há pouco, "asdasd". Significa que tudo funciona conforme esperado.

Então, deletaremos "asdasd" do HTML e, para conseguirmos mexer o conteúdo de texto contido no elemento `sugestao`, em `mailcheck-config.js` poderíamos incluir `textContent`, assim:

```
document.querySelector('#email').addEventListener('blur', function() {
  Mailcheck.run({
    email: campoEmail.value,
    domains: domains, // optional
    topLevelDomains: topLevelDomains, // optional
    secondLevelDomains: secondLevelDomains, // optional
    distanceFunction: superStringDistance, // optional
    suggested: function(suggestion) {
      sugestao.style.display = 'inline-block';
      sugestao.textContent = 'Você quis dizer: ' + suggestion.full + '?';
    }
  });
});
```

Vamos testar atualizando o navegador e usando o Console. Ao digitarmos "meunomelegal@gmail.com", é exibida a mensagem "Você quis dizer: meunomelegal@gmail.com" logo abaixo do campo de preenchimento, conforme esperado.

Para que o xis na bolinha vermelha não seja exibido antes mesmo do usuário digitar incorretamente seus dados, iremos a `index.html` e deletaremos a classe `contatoCampo--erro` da `div` correspondente ao campo de email. Feito isso, testaremos novamente no navegador.

Quando ocorrer um erro de digitação, então, teremos que colocar a classe que acabamos de deletar na `div` que se encontra acima - ou seja, a classe `contatoCampo` -, como fizemos na validação do CEP.

Também seria interessante que o campo não viesse com a borda vermelha, indicando erro, antes mesmo do preenchimento por parte do usuário. Para corrigir isto, iremos ao arquivo HTML novamente, em que encontraremos a classe `contatoCampo--validouErro` no mesmo campo referente ao email. Iremos copiá-la, deletaremos o atributo classe e, em `mailcheck-config.js`, teremos:

```
sugestao.style.display = 'inline-block';
sugestao.textContent = 'Você quis dizer: ' + suggestion.full + '?';
sugestao.parentNode.classList.add('contatoCampo--erro');
campoEmail.classList.add('contatoCampo--validouErro');
```

Neste caso, não é necessário incluir o ponto `.` antes de `contatoCampo--erro`, pois já indicamos que trata-se de uma classe com `classList.add()`.

Ao testarmos mais uma vez no navegador usando o "Tab", confirmaremos que tudo funciona como gostaríamos. Porém, repare que quando digitamos o email incorretamente e apertamos o "Tab" para pularmos para o campo seguinte, o NVDA não lê a mensagem de erro.

Ele indica que houve uma entrada inválida, porém segue para o campo de telefone sem fornecer maiores informações. Iremos consertar isto adiante.