

## Recapitulando

### Transcrição

Começando deste ponto? Você pode fazer o [DOWNLOAD \(https://github.com/alura-cursos/logica-de-programacao-I/archive/aula-2.zip\)](https://github.com/alura-cursos/logica-de-programacao-I/archive/aula-2.zip) do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

---

Antes de avançar, vamos falar sobre convenção de código. Para pensar sobre isso vamos imaginar um prédio, todos os moradores de um condomínio são submetidos a mesma convenção de regras. Por exemplo, existe uma regra que diz: ninguém pode entrar molhado no elevador. Ninguém te impedirá de entrar molhado no elevador se você quiser fazer isso, porém, isso não será bem visto por outros moradores e essa infração pode ser submetida a multa.

Em programação temos algo parecido: a convenção do código. Existe uma convenção que afirma, por exemplo, que as *tags* HTML, como a `<h1>`, devem vir em letras minúsculas. Se utilizarmos em maiúsculo ( `<H1>` ) o código continuará funcionando, mas a convenção existente estipula que é uma boa prática utilizar letras minúsculas nas *tags*.

No mundo JavaScript as regras são ainda mais rígidas. Se escrevermos `ALERT` em vez de `alert`, por exemplo:

```
ALERT("Isso sim é um programa");
```

Ao salvar o arquivo e carregar a página, o alerta não é exibido. Ao abrir o console encontramos a indicação de um erro:

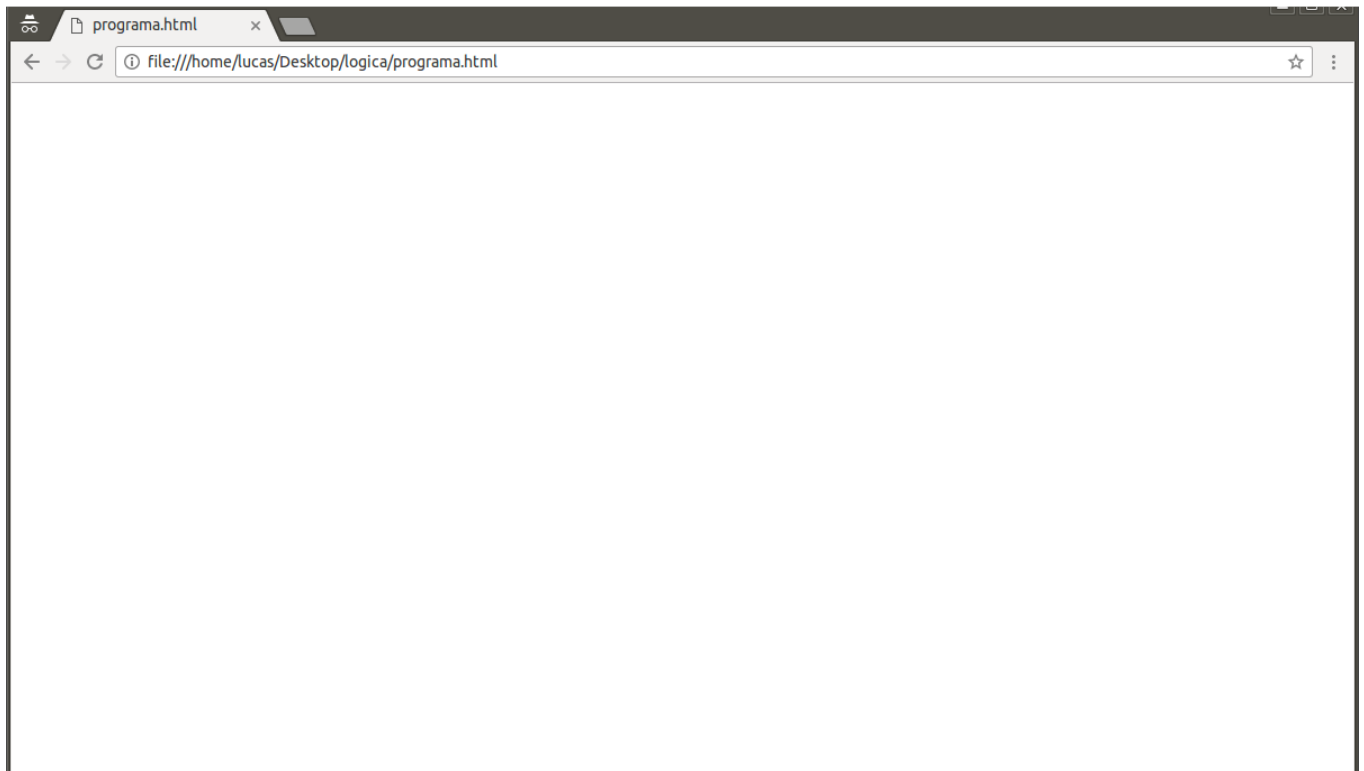
```
Uncaught ReferenceError: ALERT is not defined
```

Tradução: ALERT não está definido

Isso ocorre porque o JavaScript só está preparado para entender a sintaxe `alert` em minúsculo. Então, quando escrevemos `ALERT`, estamos cometendo um erro de sintaxe. É a mesma lógica do português, quando escrevemos uma frase sem concordância, temos um erro de sintaxe.

Após a introdução teórica vamos partir para a prática! A proposta desta aula é criar um novo programa para praticar e também aprender coisas novas. No sublime com o arquivo `primeiro_teste.html` aberto, apertamos "Ctrl + N" para criar um novo arquivo e em seguida, "Ctrl + S", para salvar o arquivo. Vamos selecionar o diretório `logica` para salvar o arquivo e nomeamos ele de `programa.html`. Vamos fechar o `primeiro_teste.html` clicando no "x" na aba do arquivo no editor. Dessa forma apenas o arquivo `programa.html` ficará aberto.

Para abrir o `programa.html` no navegador vimos os passos: menu "Arquivo > Abrir Arquivo" e em seguida navegamos até o diretório `logica` e abrimos o arquivo `programa.html`. Como ainda não programamos nada, vamos ver apenas uma tela branca:



Agora, é preciso adicionar a estrutura base do HTML e até o final do treinamento todo código que você escrever deve seguir a estrutura mínima. A ideia neste curso é que você aprenda lógica de programação, então, vamos mostrar o básico de HTML e Javascript, isto é, o mínimo que você precisa conhecer para alcançar o objetivo do curso: aprender lógica de programação.

O mínimo que deve existir no seu programa para que ele funcione é a *tag* `<meta>` acompanhada do atributo `charset` e o valor `"UTF-8"`. Um programa até funciona sem essa *tag*, mas acontecerão problemas com a acentuação dos caracteres:

```
<meta charset="UTF-8">
```

E se quisermos escrever JavaScript, o código deve estar dentro da *tag* `<script>` :

```
<meta charset="UTF-8">

<script>
</script>
```

Esse é o mínimo que precisamos para começar a programar e ter novos programas. Se salvarmos o arquivo e abrirmos no navegador nada acontece, pois não inserimos nenhum conteúdo dentro da *tag* `<script>`.

Na *tag* `<script>` vamos introduzir um `alert`, mas não utilizaremos ele "solto". Lembra que já passamos um texto para ele antes? Teremos:

```
alert("Isso sim é um programa");
```

Quando falamos sobre uma instrução que o JavaScript recebe utilizamos o termo **parâmetro**. Então, é preciso passar para o `alert` um parâmetro. Entender o que significa parâmetro faz parte da lógica de programação. Lembrando que quando escrevemos um texto no mundo JavaScript, utilizamos aspas:

```
alert "A idade do Flávio é"
```

Mas será que só isso é suficiente? Ao salvar o arquivo e recarregar a página nada acontece. Vamos verificar o que diz o Console, para acessá-lo de outra forma vamos no menu "Visualizar > Desenvolvedor > Console JavaScript". Ao acessar o console vemos que existe uma mensagem para nós: um erro na linha 4. Ao retornar para o editor podemos ver que a linha 4 é a seguinte:

```
alert "A idade do Flávio é"
```

Você lembra por que não funcionou? Porque o `alert` recebe um texto e ele deve ser passado entre parênteses:

```
alert("A idade do Flávio é")
```

Agora sim! Ao salvar o arquivo e atualizar a página no navegador vemos, finalmente, o `alert` .

Você deve estar pensando porque não colocamos ponto e vírgula no fim da linha e mesmo assim o código funcionou. Para o curso de lógica, o ponto e vírgula não fará muita diferença! Se posteriormente você quiser evoluir na linguagem JavaScript, a ausência dele pode causar problemas! Por isso, vamos desde já criar o hábito de no final de cada instrução utilizar ponto e vírgula. Dessa forma deixamos claro que temos ali o final de uma instrução:

```
alert("A idade do Flávio é");
```

