

# Dashboards com R — Módulo II

HTML e CSS para Shiny



May de 2022

HTML

# O que é?

HTML é uma **linguagem de marcação** para construir páginas web.

Uma linguagem de marcação é apenas um tipo de documento que contém texto simples (como em um bloco de notas) e um conjunto de instruções para formatar (anotar, marcar) parte específicas do conteúdo. Além do HTML, o LaTeX e o (R) Markdown são outros exemplos comuns de linguagem de marcação.

A maior parte do esforço em aprender uma linguagem de marcação está em aprender quais são e como utilizar as instruções de formatação, os seus **marcadores**.

Os marcadores no caso do HTML são as **tags**.

# Tags e elementos

As tags no HTML definem os chamados **elementos HTML**.

Um elemento HTML é composto por uma **tag de abertura**, algum **conteúdo** e uma **tag de fechamento**.

```
<nome_da_tag> conteúdo </nome_da_tag>
```

**Nota:** Alguns elementos podem aparecer corretamente sem a tag de fechamento, mas não conte sempre com isso. Resultados inesperados e erros podem acontecer se você esquecer a tag de fechamento.

# Documentos HTML

Todos os documentos HTML precisam começar com uma declaração de tipo

```
<!DOCTYPE html>
```

Ela representa o tipo de documento, o que ajuda os navegadores a mostrar as páginas corretamente. Ela deve aparecer apenas uma vez, no topo da página.

# Documentos HTML

O código HTML em si começa com a tag `<html>` e deve terminar com `</html>`.

A parte visível do documento fica entre as tags `<body>` e `</body>`.

```
<!DOCTYPE html>

<html>
  <body>
    O conteúdo da página fica aqui.
  </body>
</html>
```

**Importante:** só pode haver apenas um `<body>` em um documento HTML.

# A tag <head>

A tag <head> cria uma seção de metadados para a nossa página HTML.

O elemento <title>, por exemplo, pode ser utilizado para definir um título para a página, usado pelo navegador (na barra de ferramentas ou quando a página é adicionada aos favoritos) e por sites de busca.

```
<html>
  <head>
    <title>Esse é o título da página</title>
  </head>
  <body>
    O conteúdo da página fica aqui.
  </body>
</html>
```

Veremos outros metadados que podemos definir na seção <head> mais adiante.

# Cabeçalhos

Você pode construir títulos e subtítulos com as tags <h1> a <h6>

```
<h1>Título 1</h1>  
<h2>Título 2</h2>  
<h3>Título 3</h3>  
<h4>Título 4</h4>  
<h5>Título 5</h5>  
<h6>Título 6</h6>
```

<h1> define o título mais importante, enquanto <h6> o menos importante.

**Nota:** é uma boa prática colocar apenas um <h1> por página HTML.



# Parágrafos

Parágrafos são definidos pela tag <p>. Eles sempre iniciam em uma nova linha. Os navegadores automaticamente criam um espaço em branco (margem) antes e depois do parágrafo.

```
<p>  
  Isto é um parágrafo.  
</p>  
<p>  
  E aqui um outro parágrafo.  
</p>
```

**Nota:** O HTML não é sensível à caixa das palavras, isto é, <P> e <p> são equivalentes. No entanto, é uma boa prática utilizar a caixa baixa.

# Elementos aninhados

Elementos HTML se organizam de maneira hierárquica, isto é, os elementos são aninhados dentro de outros.

```
<!DOCTYPE html>
<html>
  <body>

    <h1>Título da página</h1>
    <p>Este é o primeiro <b>parágrafo</b> desta página HTML.</p>

  </body>
</html>
```

No código acima

- o elemento `<body>` reside dentro do elemento `html`
- os elementos `<h1>` e `<p>` residem dentro do elemento `body`
- o elemento `<b>`, que deixa o texto em negrito, reside dentro do elemento `p`.

# Atributos

Todos os elementos HTML podem ter atributos, que provêm informação adicional sobre os elementos.

Atributos são sempre especificados na tag de abertura. Eles geralmente possuem a seguinte sintaxe `nome="valor"`. Dois argumentos diferentes são separados por um espaço vazio.

```
<!DOCTYPE html>
<html lang="pt-BR">
  <body>
    "Conteúdo da página"
  </body>
</html>
```

O atributo `lang` no elemento `<html>`, por exemplo, declara qual será o idioma utilizado na página.

**Nota:** é uma boa prática declarar o idioma da página, pois isso é utilizado pelos algoritmos de busca e pelos navegadores.

# Links

Links HTML são chamados de **hiperlinks** (ou hiperligação).

Eles são criados com a tag <a>.

```
<a href="https://curso-r.com">  
  Clique aqui para acessar o site da curso-r  
</a>
```

Veja que este elemento possui o atributo href, que define o destino do link.

O conteúdo deste elemento será o texto visível na tela para ser clicado.

# Comentários

Podemos adicionar comentários ao nosso código HTML a partir da seguinte sintaxe

- use `<!--` para iniciar o comentário
- use `-->` para encerrar o comentário

```
<p>Isto é um parágrafo</p>  
<!-- Isto é um comentário e vai ser ignorado pelo navegador. -  
<p>Isto é um outro parágrafo</p>
```

# Elementos vazios

Alguns elementos HTML não possuem conteúdo e, portanto, não precisam de uma tag de fechamento. Esses elementos são chamados de **elementos vazios**.

A tag `<br>`, que gera uma quebra de linha na página, é um exemplo de elemento vazio.

```
<p>Parágrafo 1</p>  
<br> <!-- Espaço em branco -->  
<p>Parágrafo 2</p>
```

Veja que não precisamos fechar a tag `<br>` com uma tag `</br>`

# Imagens

Imagens podem ser inseridas em uma página HTML a partir da tag <img>.

```
<img src = "caminho_ou_url_da_imagem" width = "100px" height =
```

No elemento acima:

- o argumento `src` é utilizado para especificar o caminho ou URL da imagem
- os argumentos `width` e `height` são utilizados para especificar o comprimento e altura da imagem;
- especificar o comprimento e altura é importante pois o navegador reserva o espaço da imagem na tela, mantendo o layout da página, caso ela demore para ser carregada;
- podemos usar o argumento `alt` para atribuir uma descrição à imagem, o que é utilizado por leitores de tela para descrever a imagem para pessoas com deficiências visuais.

# Elementos em bloco e em linha

**Elementos em bloco** sempre começam em uma nova linha e ocupam todo o comprimento da tela ou todo o comprimento que tiverem a disposição. Os navegadores automaticamente adicionam algum espaço (margem) antes e depois desses elementos.

**Elementos em linha** não iniciam uma nova linha. Esses elementos só ocupam o comprimento necessário para apresentar seu conteúdo na tela.



# A tag <div>

A tag <div> é um elemento em bloco normalmente é utilizado como um *container* para outros elementos HTML. Se você não atribuir nenhum atributo a ela, como comprimento ou altura, ela não gera nenhum efeito na página.

Normalmente aplicamos a ela atributos que vão definir o estilo de parte ou de todos os elementos que a <div> contém.

```
<div>
  <p>Um parágrafo</p>
  <a href="https://curso-r.com">Um link</a>
</div>
```

# A tag <span>

A tag <span> é um elemento em linha utilizada como um *container* para outros elementos HTML. Se você não atribuir nenhum atributo a ela, como comprimento ou altura, ela não gera nenhum efeito na página.

Normalmente aplicamos a ela atributos que vão definir o estilo de parte ou de todos os elementos que a <div> contém.

```
<p>
```

```
  Como formatar uma única <span>palavra</span>?
```

```
</p>
```

CSS

# O que é?

O CSS (*Cascading Style Sheets*) é uma linguagem de folha de estilo utilizada para formatar o layout de uma página Web.

Com CSS, podemos controlar a cor, fonte, tamanho do texto, espaçamento entre elementos, cores e imagens de fundo, a maneira como os elementos são mostrados na página a depender dos diferentes tamanhos de tela e muito mais!

**Nota:** a palavra *cascading* (cascata) significa que um estilo aplicado a um elemento pai também é aplicado a todos os elementos filhos.

# Sintaxe

A seguir, temos uma **regra** CSS que define a cor dos parágrafos (o conteúdo dos elementos <p>) como azul e o tamanho da fonte como 12px.

```
p {  
  color: blue;  
  font-size: 12px  
}
```

No código:

- p é chamado de **seletor**, isto é, a definição dos elementos aos quais o estilo será aplicado
- color: blue; e font-size: 12px são chamadas de **declarações**, que define o estilo a ser aplicado
- color e font-size dentro de cada declaração são chamados de **propriedades**, isto é, a característica de cada elemento que será alterada.
- já blue e 12px são os **valores** atribuídos a cada propriedade.

# Usando CSS

O CSS pode ser inserido em um código HTML de 3 maneiras:

- em linha (*inline*), usando o atributo `style` dentro de elementos HTML
- interno (*internal*), usando a tag `<style>` na seção `<head>`
- externo (*external*), usando a tag `<link>` na seção `<head>` para apontar para um arquivo CSS externo.

O jeito mais comum de adicionar CSS ao HTML é utilizando um arquivo CSS externo, mas veremos aqui como utilizar as três formas.

# CSS inline

O CSS inline é utilizado para aplicar estilo a um único elemento HTML. Fazemos isso usando o atributo `style` do elemento.

```
<p style = "color: blue;"> Esse texto terá a cor azul. </p>
```

Veja que neste caso não precisamos declarar um seletor.

# CSS interno

Um CSS interno é utilizado para definir estilo para os elementos de uma única página HTML. Ele é definido na seção <head>, dentro de uma tag <style>.

```
<html>
  <head>
    <style>
      h1 {
        color: red;
      }
      p {
        color: blue;
      }
    </style>
  </head>
  <body>
    <h1> Esse título terá a cor vermelha </h1>
    <p> Esse texto terá a cor azul </p>
  </body>
</html>
```



# CSS externo

Um arquivo CSS externo é utilizado para definir o estilo de várias páginas HTML. Para associar um arquivo CSS a uma página HTML, usamos a tag `<link>` na seção `<head>`.

A tag `<link>` é utilizada para estabelecer uma relação entre o arquivo HTML e um arquivo externo. O atributo `href` recebe o caminho para o arquivo externo (nosso arquivo CSS) e o atributo `rel` estabelece qual o tipo de relação entre os arquivos.

```
<!--Arquivo CSS-->
h1 {
  color: red;
}

<!--Arquivo HTML-->
<html>
  <head>
    <link rel="stylesheet" href="custom.css">
  </head>
</html>
```

# Precedência

Quando mais de uma declaração afeta um elemento HTML, algumas regras de precedência são utilizadas para definir qual declaração surtirá efeito.

## Quanto mais perto do elemento, maior a precedência

- O CSS inline tem precedência com relação ao CSS interno e externo.
- Com relação ao CSS interno e externo, a precedência fica com o elemento (`<style>` ou `<link>`) inserido depois.
- No caso de duas declarações com o mesmo seletor e para a mesma propriedade, mas com valores diferentes, a segunda declaração terá precedência sobre a primeira.

**Nota:** existem outras regras de precedência com relação à **especificidade do seletor**. Para mais informações, [veja este artigo da W3Schools](#) em inglês ou [este artigo](#) em português.

# Cores

Para alterar cores, utilizamos as propriedades `color` e `background-color`.

```
p {  
  color: white;  
}  
  
h1, h2 {  
  color: blue;  
}  
  
body {  
  background-color: black  
}
```

**Nota:** repare que você pode selecionar um grupo de elementos separando cada seletor por uma vírgula (`h1, h2`).

# Tamanho da fonte

Para alterar o tamanho da fonte, utilizamos a propriedade `font-size`.

```
p {  
  font-size: 10px;  
}  
  
h1 {  
  font-size: 12pt;  
}
```

# Fontes

Para alterar a fonte do texto, utilizamos a propriedade `font-family`.

```
p {  
  font-family: "Times New Roman", Times, serif;  
}  
  
p {  
  font-family: Arial, Helvetica, sans-serif;  
}
```

**Nota:** você pode especificar mais de uma fonte em uma declaração, o que é útil para garantir que pelo menos uma das fontes especificadas esteja instalada no navegador de quem acessar a página HTML. A preferência é da esquerda para a direita.

**Recomendação:** leia as [seções sobre fonte do W3Schools](#).

# Usando fontes do Google fontes

Para importar uma fonte do Google fontes, basta utilizar o elemento `<link rel = "stylesheet">` passando o link da fonte no atributo href.

```
<head>
  <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Sofia" />
  <style>
    body {
      font-family: "Sofia", sans-serif;
    }
  </style>
</head>
```

Link para a fonte Sofia: <https://fonts.google.com/specimen/Sofia?query=safia>

# Bordas

Para definir uma borda a um elemento, utilizamos a propriedade `border`.

```
h1 {  
  border: 1px solid black;  
}
```

- O valor `1px` se refere à espessura da borda.
- O valor `solid` se refere ao estilo da borda.
- O valor `black` se refere à cor da borda.

**Nota:** você pode definir uma borda para quase todos os elementos HTML.

# Classes

Elementos HTML possuem um atributo `class` que pode ser utilizado como seletor em declarações CSS.

```
<!-- HTML -->
<p class = "azul">Este parágrafo ficará azul</p>
<p>Este parágrafo não ficará azul</p>

<!-- CSS -->
.azul {
  color: blue;
}
```

Repare que para usar uma classe como seletor, colocamos um `.` antes do nome da classe.

**Nota:** elementos HTML podem ter mais de uma classe. Múltiplas classes são separadas por um espaço. Uma mesma classe pode ser utilizada em mais de um elemento HTML.



# Ids

Elementos HTML possuem um atributo `id` que pode ser utilizado como seletor em declarações CSS. Ao contrário das classes, o `id` deve ser único dentro do documento HTML, isto é, dois elementos HTML não devem ter o mesmo `id`.

```
<!-- HTML -->
<p id = "paragrafoAzul">Este parágrafo ficará azul</p>
<p>Este parágrafo não ficará azul</p>

<!-- CSS -->
#paragrafoAzul {
    color: blue;
}
```

Repare que para usar um `id` como seletor, colocamos um `#` antes do nome do `id`.

# Margem

Margens são utilizadas para criar espaço em branco entre elementos HTML.

```
p {  
  margin: 1px 2px 4px 3px;  
}  
  
p {  
  margin-top: 1px;  
  margin-right: 2px;  
  margin-bottom: 4px;  
  margin-left: 3px;  
}
```

As duas regras CSS acima são equivalentes.

# Padding

A propriedade `padding` é utilizada para criar espaço ao redor do conteúdo de um elemento, dentro de qualquer borda definida.

```
p {  
  padding: 1px 2px 4px 3px;  
}
```

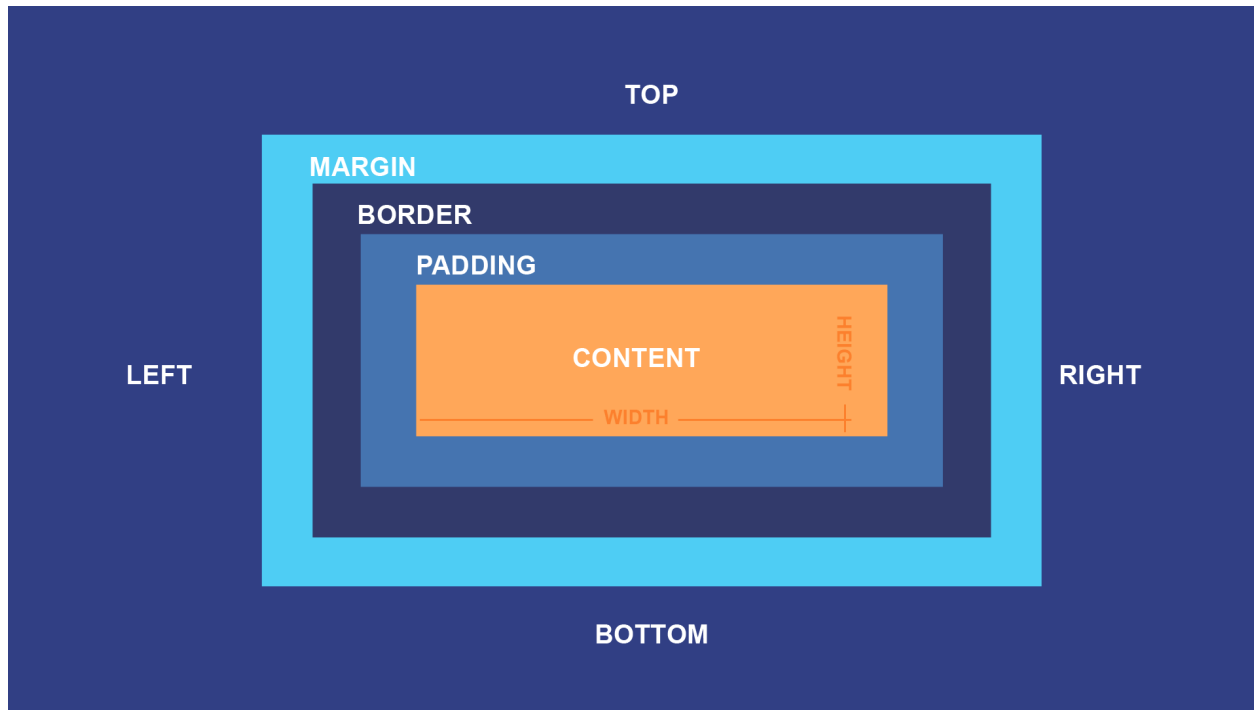
```
p {  
  padding-top: 1px;  
  padding-right: 2px;  
  padding-bottom: 4px;  
  padding-left: 3px;  
}
```

As duas regras CSS acima são equivalentes.

# Box model

O *box model* do CSS é essencialmente uma caixa que envolve todos os elementos HTML. Ele consiste de uma margem, uma borda, *padding* e o próprio conteúdo do elemento.

A imagem abaixo ilustra o *box model*:



# Display

A propriedade `display` especifica se e como um elemento HTML é exibido na tela.

Todo elemento HTML tem um valor padrão para essa propriedade. O valor padrão do `display` para a maioria dos elementos é `block` (em bloco) ou `inline` (em linha). Utilizando essa propriedade, podemos fazer uma `<div>` em linha ou um `<img>` em bloco.

```
#divInline {  
  display: inline  
}  
  
#imgBlock {  
  display: block;  
}
```

**Nota:** também é possível atribuir o valor `none` a essa propriedade, fazendo com que o elemento não seja mostrado na tela. Isso é utilizado com JavaScript para mostrar/esconder elementos.

# Posicionamento

A propriedade `position` define que tipo de posicionamento será usado em um elemento HTML.

As principais opções são:

- `static`, elementos são renderizados na ordem que eles aparecem no documento HTML (valor padrão)
- `relative`, os elementos são posicionados relativamente a sua posição normal, permitindo o ajuste do elemento a partir de `offsets`
- `absolute`, os elementos são posicionados relativamente ao seu primeiro elemento ancestral com `position` diferente de `static`
- `fixed`, os elementos são posicionados relativamente à janela do navegador

# Atividade

Vamos utilizar o que aprendemos de HTML e CSS para personalizar o visual de um aplicativo Shiny.



Ao RStudio: .R

# Exercícios

- 1) O que são elementos HTML?
- 2) Para que serve a seção <head> de um código HTML?
- 3) Conserte o erro do código HTML abaixo:

```
<p> Para acessar a loja, basta clicar <a href = "https://loja.
```

- 4) Qual a diferença entre as tags <div> e <span>?
- 5) Ao que se refere o termo *cascading* da sigla CSS?
- 6) Qual a ordem de precedência com relação ao local onde CSS pode ser colocado (inline, interno, externo)?
- 7) O que é o *box model* no CSS?



# Exercícios

8) Segundo o código CSS abaixo, a caixa azul vai aparecer sobre o fundo amarelo ou rosa? E a caixa laranja?

```
<div style = "position: relative;">
  <div style = "height: 100px; background-color: yellow;"></div>
  <div style = "height: 100px; background-color: pink;">
    <div style = "height: 40px; width: 10%; background-color:
      Caixa azul
    </div>
    <div style = "height: 40px; width: 10%; background-color:
      Caixa laranja
    </div>
  </div>
</div>
```

9) Reproduza a página HTML contida [neste link](#). Sinta-se livre para deixar o visual da página da sua maneira.

# Referências e material extra

Estes slides são um resumo adaptado e traduzido do tutorial da w3schools.

- [Tutorial de HTML da w3schools](#)
- [Tutorial de CSS da w3schools](#)