



*DÊ UM START EM
SUA CARREIRA DEV.*

PARTE 2

MÓDULO C - CSS AVANÇADO

AULA 1 - O BOX MODEL

Todo iniciante deve primeiro começar com o básico. No caso do CSS, o básico é aprender o box model. Antes de prosseguir com o aprendizado de quaisquer outros conceitos CSS, este é o que você deve dominar primeiro!

Isso tende a causar confusão com os desenvolvedores iniciantes, então agora é o momento de esclarecer tudo. Esperançosamente, esta aula irá ajudá-lo a aprender todos os elementos básicos do modelo de caixa(box model) e como eles estão conectados.

Antes de mergulhar mais fundo, todos precisam entender que cada elemento no HTML é uma caixa retangular. Você provavelmente já ouviu isso várias vezes antes, mas este é um conceito importante que todo desenvolvedor deve estar ciente.

Quando um navegador renderiza(desenha) uma página da web, cada elemento, por exemplo, um pedaço de texto ou uma imagem, é desenhado como uma caixa retangular

seguindo as regras do CSS Box Model.

No centro da caixa está o próprio conteúdo, que ocupa uma certa altura e largura. Esta região é conhecida como área de conteúdo.

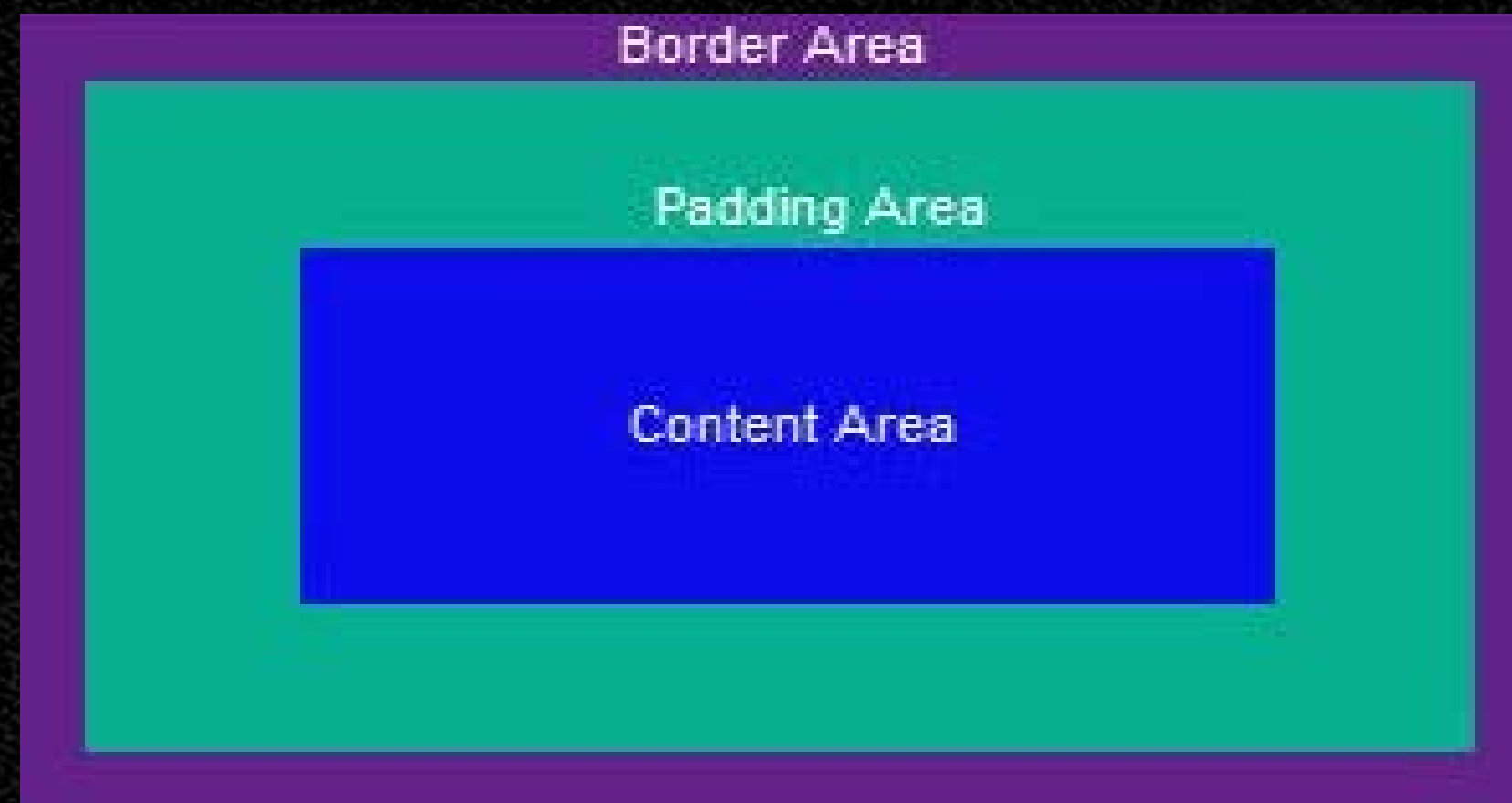
O tamanho da área de conteúdo pode ser determinado automaticamente ou você pode definir explicitamente o tamanho da altura e da largura.



Em torno da área de conteúdo, esta é uma região conhecida como área de preenchimento(padding). O tamanho do preenchimento (padding) pode ser o mesmo ao redor, ou você pode definir individualmente para preenchimento superior, inferior, esquerdo e direito. Se você estiver usando um plano de fundo para o elemento, o plano de fundo se estenderá até a área de preenchimento.

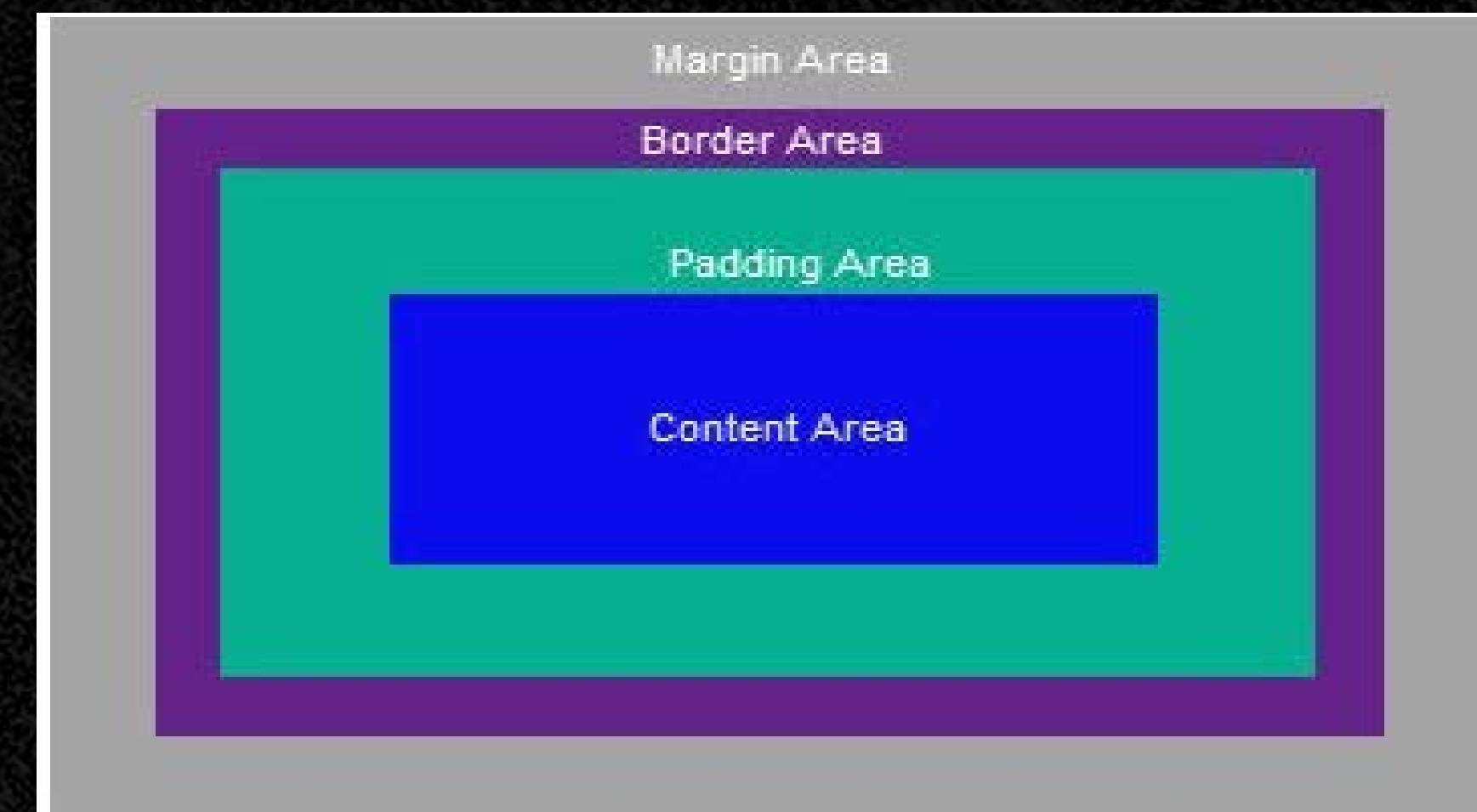


Em seguida, há uma área de fronteira(borda). Isso cria uma borda ao redor do elemento e seu preenchimento. Você pode definir a espessura, cor e estilo da borda. As opções de estilo incluem nenhum, sólido, tracejado, pontilhado e vários outros.



Finalmente, existe a área marginal(margin). Isso cria um

espaço livre ao redor do elemento, preenchimento e borda. Novamente, você pode definir individualmente as margens superior, inferior, esquerda e direita. Sob certas circunstâncias, ocorre o colapso das margens e as margens entre os elementos adjacentes podem ser compartilhadas.



Leia (em inglês): [The box model](#)

EXERCÍCIO - CRIANDO BOX

Nessa atividade vamos praticar um pouco sobre os conceitos de box model.

Crie 3 box um embaixo do outro, seguindo o exemplo abaixo. Você vai utilizar as seguintes propriedades, quanto aos valores passados, fica a seu critério, use a criatividade:

1. width
2. height
3. background

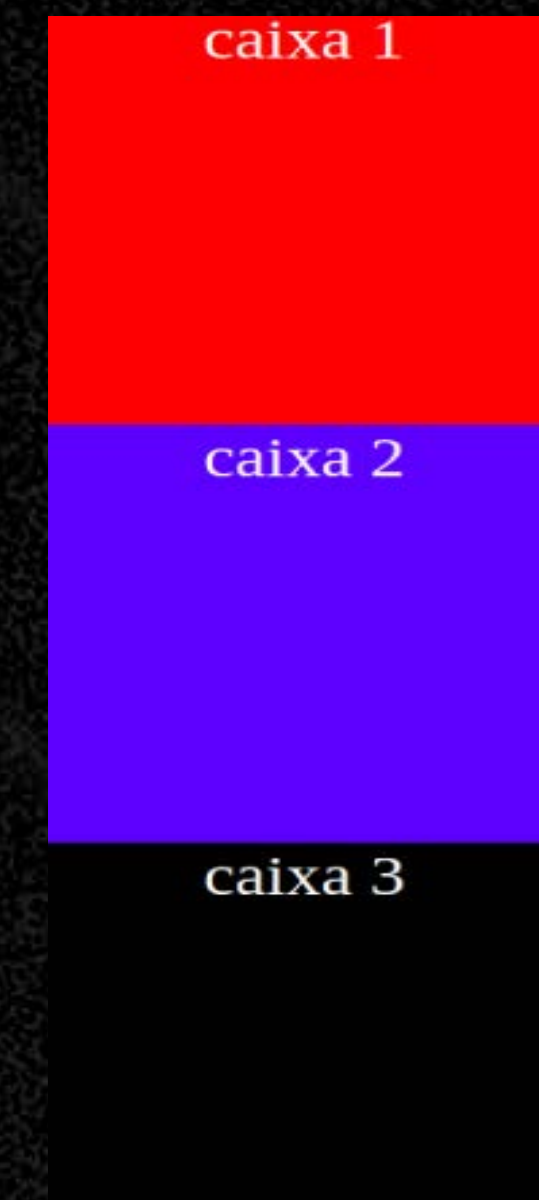
Sugestão (cores utilizadas no exemplo):

Caixa 1: #ef482a

Caixa 2: #6e71fc

Caixa 3: #000000

4. color
5. font-size
6. text-align



Acesse o [exercício](#)

AULA 2 - PADDING, MARGIN E BORDA

PADDING E MARGIN

Existem muitas maneiras de fornecer espaçamento entre os elementos e o conteúdo. Dois deles são preenchimento (padding) e margem.

Padding (preenchimento)

O preenchimento (padding), conforme mencionado anteriormente no Box Model, é o espaçamento entre o conteúdo real e a borda da caixa.

Existem diferentes maneiras de escrever os valores da propriedade padding.




```
div {  
  padding: 10px; /* Todos os Lados */  
  padding: 10px 5%; /* Vertical Horizontal */  
  padding: 10px 20px 30em; /* Top Horizontal Bottom */  
  padding: 10px 20px 30rem 40px; /* acima 1em padding */  
    /* direita 3px padding */  
    /* abaixo 30px padding */  
    /* esquerda 5px padding */  
  
  padding-left: 10px; /* Esquerda */  
  padding-right: 10px; /* Direita */  
  padding-top: 10px; /* Top */  
  padding-bottom: 10px; /* Bottom */  
}
```

O padding também possui valores especiais que podem ser usados.

?: porcentagem em relação à largura do contêiner

initial: o padrão é 0

inherit: valor de preenchimento do elemento que o contém

unset: remove o valor

Mas temos que ficar atentos ao tipo de display dos nossos elementos, pois há diferença de aplicação para cada um deles

(Inline | Block | Inline Block | Grid | Flex)

Margin (margem)

A margem (margin), conforme mencionado anteriormente no Box Model, é o espaçamento entre um elemento e outro.

Existem diferentes maneiras de escrever os valores da propriedade margin.

```
div {  
  margin: 10px; /* Todos os Lados */  
  margin: 10px 5%; /* Vertical Horizontal */  
  margin: 10px 20px 30em; /* Top Horizontal Bottom */  
  margin: 10px 20px 30rem 40px; /* Top Direita Bottom Esquerda */  
  
  margin-left: 10px; /* Esquerda */  
  margin-right: 10px; /* Direita */  
  margin-top: 10px; /* Top */  
  margin-bottom: 10px; /* Bottom */  
}
```


Mais Exemplos

```
div{
```

```
margin: 5%;
```

```
/* todos os lados: margem de 5% */
```

```
margin: 10px;
```

```
/* todos os lados: margem de 10px */
```

```
margin: 1.6em 20px;
```

```
/* topo e inferior: margem de 1.6em */  
/* esquerda e direita: margem de 20px */
```

```
margin: 10px 3% 1em;
```

```
/* topo: margem de 10px */  
/* esquerda e direita: margem de 3% */  
/* inferior: margem de 1em */
```

```
margin: 10px 3px 30px 5px; /* topo: margem de 10px */  
/* direita: margem de 3px */  
/* inferior: margem de 30px */  
/* esquerda: margem de 5px */
```



```
margin: 2em auto;
```

```
/* topo e inferior: margem de 2em */  
/* caixa está horizontalmente centralizada */
```

```
margin: auto;  
}
```

```
/* topo e inferior: margem de 0 */  
/* caixa está horizontalmente centralizada */
```

Referências

Leia: [Margin - w3schools.](#)

Leia: [Padding - w3schools.](#)

Leia: [Margin - mdn.](#)

Leia: [Padding - mdn.](#)

BORDER (borda)

A propriedade CSS border é usada para definir a borda de um elemento HTML.

A propriedade border é uma abreviação para três subpropriedades que definem o estilo, a cor e a largura de uma borda.

Exemplo:

```
border: 1px solid red;  
border: (espessura) (tipo) (cor);
```

Estilo da borda

Define o estilo da borda. O padrão é none (nenhuma borda).

```
border-style: none | hidden | dotted |
```

```
dashed | solid | double | groove | ridge |  
inset | outset
```

Cor da borda

Define a cor da borda. O padrão é ocurrentColor, definido pela propriedade color do elemento.

```
border-color: red;
```

Bordas individuais em CSS

As subpropriedades e propriedades da borda também podem ser aplicadas a um lado individual de um elemento da web.

```
border-left: green;  
border-top: pink;  
border-right: blue;  
border-bottom: skyblue;
```




Leia: [Bordas - w3schools.](#)

Leia: [Bordas - mdn.](#)



Ferramentas

Experimente!: [Border Generator.](#)

Experimente!: [9elements.](#)

Experimente!: [CSS Border Generator.](#)

Experimente!: [Border-image generator.](#)

AULA 3 - COLUMNS (MÚLTIPLAS COLUNAS DE TEXTO)

A propriedade Columns estende o modo de layout de bloco para permitir a fácil definição de múltiplas colunas de texto.

As pessoas têm problemas para ler texto se as linhas forem muito longas; se levar muito tempo para os olhos se moverem do final de uma linha para o início da seguinte, eles perderão o controle da linha em que estavam. Portanto, para aproveitar ao máximo uma tela grande, os autores devem

ter colunas de texto de largura limitada colocadas lado a lado, como fazem os jornais.



Exemplo:

Utilizando Columns CSS

WELL, THAT'S THE TRICK, ISN'T IT? AND IT'S GOING TO COST YOU SOMETHING EXTRA. TEN THOUSAND IN ADVANCE. TEN THOUSAND? WE COULD ALMOST BUY OUR OWN SHIP FOR THAT! BUT WHO'S GOING TO FLY IT, KID! YOU?

You bet I could. I'm not such a bad pilot myself! We don't have to sit here and listen... We haven't that much with us. But we could pay you two thousand now, plus fifteen when we reach Alderaan. Seventeen, huh! Okay. You guys got yourself a ship. We'll leave as soon as you're ready. Docking bay Ninety-four. Ninety-four. Looks like somebody's beginning to take an

I hope the old man got that tractor beam out of commission, or this is going to be a real short trip. Okay, hit it! We're coming up on the sentry ships. Hold 'em off! Angle the deflector shields while I charge up the main guns! I can't believe he's gone. There wasn't anything you could have done. Come on, buddy, we're not out of this yet! You in, kid? Okay, stay sharp!

an interpreter, and not very good at telling stories. Well, not at making them interesting, anyways. Well, my little friend, you've got something jammed in here real good. Were you on a cruiser or...

Infelizmente, isso era impossível de fazer com CSS e HTML sem forçar quebras de coluna em posições fixas, ou restringir severamente a marcação permitida no texto, ou usar scripts heróicos. Essa limitação é resolvida com a adição de novas propriedades CSS para estender o modo de layout de bloco tradicional.

Usando colunas

A column-count propriedade define o número de colunas para um número específico. Por exemplo,

```
<div id="col">  
  <p>  
    Lorem ipsum dolor sit amet,  
    consectetur adipiscing elit,
```

```
    sed do eiusmod tempor incididunt ut  
    labore et dolore magna  
    aliqua.
```

```
</p>
```

```
<p>
```

```
  Ut enim ad minim veniam, quis nostrud  
  exercitation ullamco
```

```
  laboris nisi ut aliquip ex ea commodo  
  consequat.
```

```
</p>
```

```
<p>
```

```
  Duis aute irure dolor in reprehenderit in  
  voluptate velit
```

```
    esse cillum dolore eu fugiat nulla  
  pariatur.
```

```
</p>
```

```
<p>
```

```
  Excepteur sint occaecat cupidatat non
```


proident, sunt in
culpa qui officia deserunt mollit anim
id est laborum.

```
</p>  
</div>
```

CSS

```
#col {  
  column-count: 2;  
}
```

Resultado

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Outras propriedades

column-count
column-gap
column-rule-style
column-rule-width
column-rule-color
column-rule
column-span
column-width



Veja mais nesse link

Leia: [CSS columns Property - w3schools](https://www.w3schools.com/css/css_columns.asp)

EXERCÍCIO - CRIANDO UM TEXTO COM COLUMNS

Nessa atividade vamos praticar um pouco sobre os conceitos de COLUMNS. Crie um texto com três ou mais parágrafos e separe eles em três ou mais colunas

Utilizando Columns CSS

WELL, THAT'S THE TRICK, ISN'T IT? AND IT'S GOING TO COST YOU SOMETHING EXTRA. TEN THOUSAND IN ADVANCE. TEN THOUSAND? WE COULD ALMOST BUY OUR OWN SHIP FOR THAT! BUT WHO'S GOING TO FLY IT, KID! YOU?

You bet I could. I'm not such a bad pilot myself! We don't have to sit here and listen... We haven't that much with us. But we could pay you two thousand now, plus fifteen when we reach Alderaan. Seventeen, huh! Okay. You guys got yourself a ship. We'll leave as soon as you're ready. Docking bay Ninety-four. Ninety-four. Looks like somebody's beginning to take an

I hope the old man got that tractor beam out of commission, or this is going to be a real short trip. Okay, hit it! We're coming up on the sentry ships. Hold 'em off! Angle the deflector shields while I charge up the main guns! I can't believe he's gone. There wasn't anything you could have done. Come on, buddy, we're not out of this yet! You in, kid? Okay, stay sharp!

an interpreter, and not very good at telling stories. Well, not at making them interesting, anyways. Well, my little friend, you've got something jammed in here real good. Were you on a cruiser or...



Acesse o [exercício](#)

AULA 4 - FLEXBOX

Neste módulo, você aprenderá como usar o Modelo de Caixa Flexível, o Flexbox, no CSS para controlar o layout de elementos em uma página web.

Flexbox é um método de layout unidimensional para o layout de itens em linhas ou colunas. Os itens flexionam para preencher o espaço adicional e encolhem para caber em espaços menores.

Por muito tempo, as únicas ferramentas compatíveis entre navegadores confiáveis disponíveis para a criação de layouts CSS eram coisas como float e position. Estes são bons e funcionam,

mas em alguns aspectos também são bastante limitantes e frustrantes.

Flex Container

O Flex Container é a tag que envolve os itens flex, ao indicar display: flex, essa tag passa a ser um Flex Container.

display

Define o elemento como um flex container, tornando os seus filhos flex-itens.

display: flex;

/*Torna o elemento um flex container automaticamente transformando todos os seus filhos diretos em flex itens.*/

[Veja o exemplo](#)

flex-direction

Define a direção dos flex itens. Por padrão ele é row (linha), por isso quando o display: flex; é adicionado, os elementos ficam em linha, um do lado do outro.

A mudança de row para column geralmente acontece quando estamos definindo os estilos em media queries para o mobile. Assim você garante que o conteúdo seja apresentado em coluna única.

flex-direction: row;

/* Os itens ficam em linha*/

flex-direction: row-reverse;

/*Os itens ficam em linha reversa, ou seja 3, 2, 1.*/

flex-direction: column;

/*Os itens ficam em uma única coluna, um embaixo do outro.*/

flex-direction: column-reverse;

/*Os itens ficam em uma única coluna, um embaixo do outro, porém em ordem reversa: 3, 2 e 1.*/

[Veja o exemplo](#)

flex-wrap

Define se os itens devem quebrar ou não a linha. Por padrão eles não quebram linha, isso faz com que os flex itens sejam compactados além do limite do conteúdo.

Essa é geralmente uma propriedade que é quase sempre definida como flex-wrap: wrap; Pois assim quando um dos flex itens atinge o limite do conteúdo, o último item passa para a coluna debaixo e assim por diante.

flex-wrap: nowrap;

/*Valor padrão, não permite a quebra de linha.*/

flex-wrap: wrap;

/* Quebra a linha assim que um dos flex

itens não puder mais ser compactado.*/

flex-wrap: wrap-reverse;

/* Quebra a linha assim que um dos flex itens não puder mais ser compactado. A quebra é na direção contrária, ou seja para a linha acima.*/

[Veja o exemplo](#)

flex-flow

O flex-flow é um atalho para as propriedades flex-direction e flex-wrap. Você não verá muito o seu uso, pois geralmente quando mudamos o flex-direction para column, mantemos o padrão do flex-wrap que é nowrap.

E quando mudamos o flex-wrap para

wrap, mantemos o padrão do flex-direction que é row.

flex-flow: row nowrap;

/*Coloca o conteúdo em linha e não permite a quebra de linha.*/

flex-flow: row wrap;

/*Coloca o conteúdo em linha e permite a quebra de linha.*/

flex-flow: column nowrap;

/*Coloca o conteúdo em coluna e não permite a quebra de linha.*/

[Veja o exemplo](#)

justify-content

Alinha os itens flex no container de acordo com a direção. A propriedade só

funciona se os itens atuais não ocuparem todo o container. Isso significa que ao definir flex: 1; ou algo similar nos itens, a propriedade não terá mais função

Excelente propriedade para ser usada em casos que você deseja alinhar um item na ponta esquerda e outro na direita, como em um simples header com marca e navegação.

justify-content: flex-start;

/*Alinha os itens ao início do container.*/

justify-content: flex-end;

/*Alinha os itens ao final do container.*/

justify-content: center;

/*Alinha os itens ao centro do container.*/

`justify-content: space-between;`
/*Cria um espaçamento igual entre os elementos.
Mantendo o primeiro grudado no início e o último no final.*/

`justify-content: space-around;`
/*Cria um espaçamento entre os elementos.
Os espaçamentos do meio são duas vezes maiores que o inicial e final.*/

[Veja o exemplo](#)

`align-items`

O align-items alinha os flex itens de acordo com o eixo do container. O alinhamento é diferente para quando os itens estão em colunas ou linhas.

Essa propriedade permite o tão sonhado alinhamento central no eixo vertical, algo que antes só era possível com diferentes hacks.

`align-items: stretch;`
/*Valor padrão, ele que faz com que os flex itens cresçam igualmente.*/

`align-items: flex-start;`
/*Alinha os itens ao início.*/

`align-items: flex-end;`
/*Alinha os itens ao final.*/

`align-items: center;`
/*Alinha os itens ao centro.*/

`align-items: baseline;`
/*Alinha os itens de acordo com a linha

base da tipografia.*/*

[Veja o exemplo](#)

align-content

Alinha as linhas do container em relação ao eixo vertical. A propriedade só funciona se existir mais de uma linha de flex-itens. Para isso o flex-wrap precisa ser wrap.

Além disso o efeito dela apenas será visualizado caso o container seja maior que a soma das linhas dos itens. Isso significa que se você não definir height para o container, a propriedade não influencia no layout.

`align-content: stretch;`

`/*Valor padrão, ele que faz com que os flex itens cresçam igualmente na vertical.*/*`

`align-content: flex-start;`

`/*Alinha todas as linhas de itens ao início.*/*`

`align-content: flex-end;`

`/*Alinha todas as linhas de itens ao final.*/*`

`align-content: center;`

`/*Alinha todas as linhas de itens ao centro.*/*`

`align-content: space-between;`

`/*Cria um espaçamento igual entre as linhas.`

`Mantendo a primeira grudada no topo e a última no bottom.*/*`


```
align-content: space-around;  
/*Cria um espaçamento entre as linhas.  
Os espaçamentos do meio são duas  
vezes maiores que o top e bottom.*/
```

[Veja o exemplo](#)

➡ Para Saber mais sobre o flex, veja mais exemplos nesse link:
Conheça!: [Coding Imweb](#)

Flexbox

➡ Leia: [Flexbox - MDN](#)
Leia: [Flexbox - w3schools](#)
Leia: [Flexbox - Flexbox guia completo em inglês](#)

EXERCÍCIO - UTILIZANDO FLEX

Nessa atividade vamos praticar um pouco mais sobre os conceitos de Flexbox.

Exercício

Aplicar display flex

Alinhar ao centro da altura do Header

Escolher alguma fonte para seu projeto

Mudar cor do texto e link

Remover a estilização da lista

[Acesse o exercício](#)

AULA 5 - RESPONSIVIDADE

O design responsivo da Web é a abordagem que sugere que o design e o desenvolvimento devem responder ao comportamento do usuário e ao ambiente com base no tamanho da tela, plataforma e orientação.

A prática consiste em uma mistura de grades e layouts flexíveis, imagens e um uso inteligente de media queries CSS. Conforme o usuário muda do laptop para o iPad, o site deve mudar automaticamente para acomodar a resolução.

É importante entender que o web

design responsivo não é uma tecnologia separada - é um termo usado para descrever uma abordagem ao web design ou um conjunto de práticas recomendadas, usado para criar um layout que pode responder ao dispositivo que está sendo usado para visualizar o conteúdo.

Defina a janela de visualização

As páginas otimizadas para uma variedade de dispositivos devem incluir uma meta tag viewport no cabeçalho do documento. Uma meta tag viewport fornece ao navegador instruções sobre como controlar as dimensões e o

dimensionamento da página.

```
<meta name="viewport"  
content="width=device-width, initial-  
scale=1">
```

A propriedade width controla o tamanho da viewport. O valor desta propriedade pode ser definido com um número específico de pixels como por exemplo width=600 ou com um valor especial chamado device-width que representa a largura da onde o conteúdo está sendo apresentado em pixels de CSS considerando uma escala de 100%. (Ainda há as propriedades height e device-height, as quais podem ser úteis para páginas com elementos que mudam de posição baseado na altura da viewport.)

A propriedade initial-scale controla o nível de amplificação quando a página é carregada pela primeira vez. As propriedades maximum-scale, minimum-scale, e user-scalable controlam a permissão para o usuário efetuar aumento ou diminuição da página.

Viewport

Viewport - janela de visualização

A janela de visualização é a área visível do usuário em uma página da web.

A janela de visualização varia com o dispositivo e será menor em um telefone celular do que em uma tela de computador.

Antes dos tablets e telefones celulares, as páginas da web eram projetadas apenas

para telas de computador, e era comum que as páginas tivessem um design estático e um tamanho fixo.

Então, quando começamos a navegar na Internet usando tablets e telefones celulares, as páginas da web de tamanho fixo eram muito grandes para caber na janela de visualização. Para corrigir isso, os navegadores nesses dispositivos reduziram a página da web inteira para caber na tela.



Links alternativos

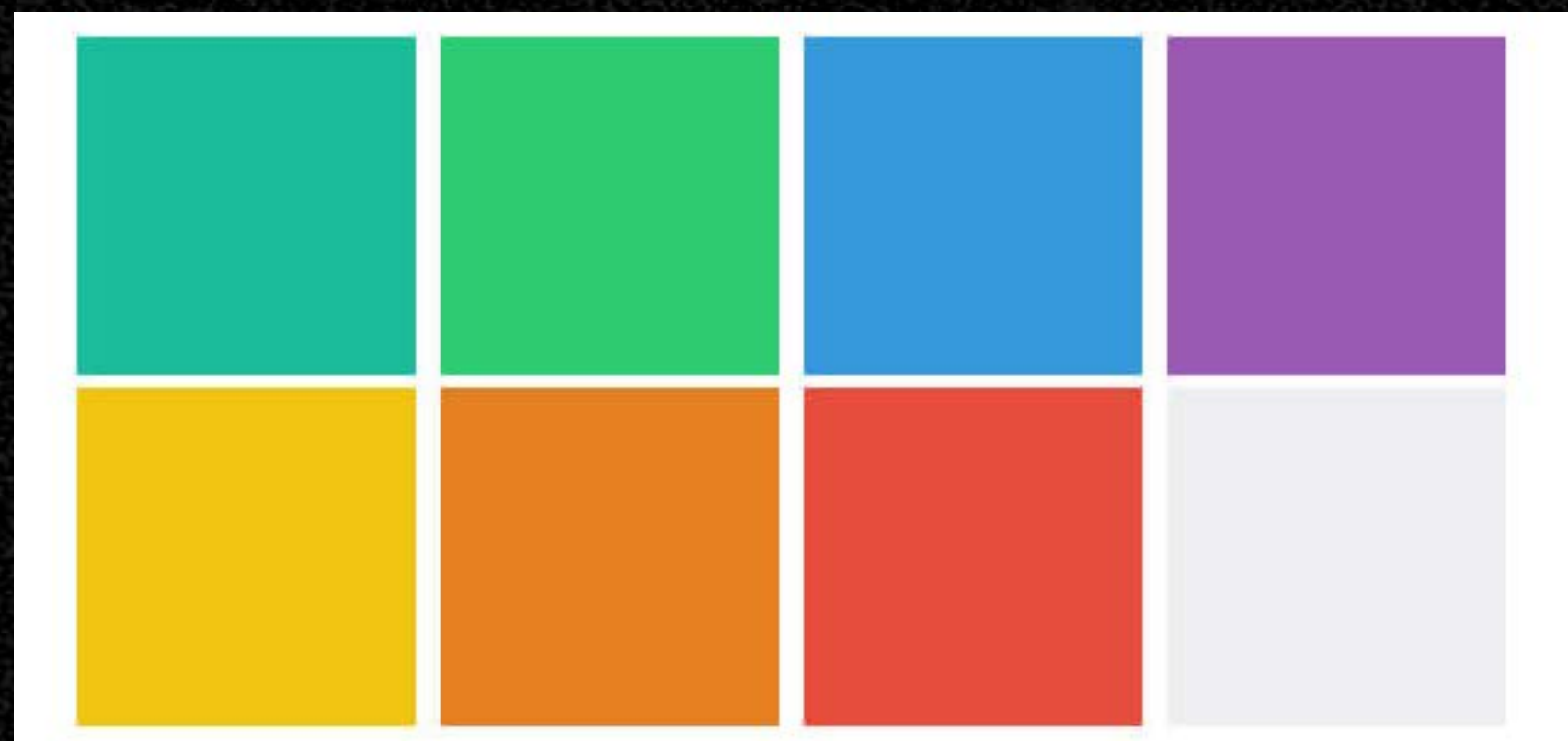
[Dev Media](#)

EXERCÍCIO - RESPONSABILIDADE

Nessa atividade vamos praticar um pouco sobre os conceitos de responsividade.

[Acesse o exercício](#)

Crie dois ou mais quadrados ou retângulos e ao alterar o tamanho da tela os mesmos devem continuar formatados.



AULA 6 - MEDIA QUERIES

Fazer um site com um layout adaptável é chamado de Web Design Responsivo. E as media query CSS são uma das partes mais importantes do design responsivo. Neste artigo, vamos dar uma olhada mais de perto nas media queries e como usá-las em CSS.

O QUE É UMA MEDIA QUERY?

Media query um recurso CSS3 que faz com que uma página da Web adapte seu layout a diferentes tamanhos de tela e tipos de mídia.

Sintaxe

```
@media media type and (condition:  
breakpoint) {
```

```
}
```

```
@media tipo de mídia && (condição:  
ponto de interrupção) {
```

```
}
```

Podemos direcionar diferentes tipos de mídia sob uma variedade de condições. Se a condição e / ou os tipos de mídia atenderem, as regras dentro da media query serão aplicadas, caso contrário,

não.

@Regra de mídia

Começamos definindo media query a regra @media e depois incluímos regras CSS entre chaves. A regra @ media também é usada para especificar os tipos de mídia de destino.

```
@media () {  
  
}
```

Parêntese

Dentro dos parênteses, definimos uma condição. Por exemplo, desejo aplicar um tamanho de fonte maior para dispositivos móveis. Para fazer isso, precisamos definir uma largura máxima que verifica a largura de um dispositivo:

```
.text {  
  font-size: 14px;  
}
```

```
@media (max-width: 480px) {  
  .text {  
    font-size: 16px;  
  }  
}
```

Normalmente, o tamanho do texto será de 14px. No entanto, como aplicamos uma media query, ela mudará para 16px quando um dispositivo tiver uma largura máxima de 480px ou menos.

Importante: Sempre coloque suas media queries no final de seu arquivo CSS.

Pontos de interrupção comuns: existe uma resolução padrão?

Uma das perguntas mais frequentes é "Qual ponto de interrupção devo usar?". Há uma tonelada de dispositivos no mercado, então não podemos e não devemos definir pontos de interrupção fixos para cada um deles.

usados na programação diária. Se você estiver usando uma estrutura CSS (como Bootstrap, Bulma, etc.), você também pode usar seus pontos de interrupção.

- 320px - 480px: Dispositivos móveis
- 481 px - 768 px: iPads, tablets
- 769 px - 1024 px: telas pequenas, laptops
- 1.025 px - 1.200 px: desktops, telas grandes

1201 px e mais - telas extragrandes, TV

Como eu disse acima, esses pontos de interrupção podem ser diferentes e não há um padrão definido exatamente, mas esses são alguns dos mais usados.

Links de referências:

[Tableless Dev Media](#)

[MDN Web](#)

EXERCÍCIO - MEDIA QUERIES

Nessa atividade vamos praticar um pouco sobre os conceitos de responsividade.

Crie uma tela e coloque uma cor de fundo quando a tela estiver com 800 pixels de largura, altere a cor de fundo quando a tela estiver entre 400 e 799 pixels de largura. Se a tela estiver com menos de 400 pixels, altere a cor de novo.

[Acesse o exercício](#)

AULA 7 - MOBILE FIRST

Nessa aula nós vamos falar sobre o conceito de mobile first.

INTRODUÇÃO

Como o próprio nome já diz, mobile first, significa “primeiro o mobile”. Ou seja, começar o desenvolvimento do front pelo layout para dispositivos móveis. E então, utilizar media queries para sobrescrever esse estilo em telas maiores, porém utilizando o min-width, ao invés do max-width.

RESPONSIVIDADE X MOBILE FIRST

É muito importante entender que fazer uma página responsiva e uma página em mobile first são duas coisas muito diferentes!

Na responsividade você apenas adapta os elementos para que eles encaixem na tela sem quebrar a página.

Com o mobile first, você vai ter dois layouts diferentes um para o desktop e outro para dispositivos móveis, podendo assim ter uma melhor disposição de elementos em cada uma das situações, de uma maneira que faça mais sentido para

cada uma delas.



Para saber mais

Leia: [Entenda o que é mobile first e conheça as suas principais vantagens](#)

Leia: [Como escrever seu css para projetos mobile-first](#)

AULA 8 - VARIÁVEIS

Variáveis CSS, mais precisamente conhecidas como propriedades personalizadas no CSS, são úteis para reduzir a repetição em CSS e também para efeitos de tempo de execução poderosos, como troca de tema e potencialmente estender futuros recursos CSS.

As propriedades personalizadas adicionam dois novos recursos à nossa caixa de ferramentas CSS:

A capacidade de você atribuir valores arbitrários a uma propriedade com um nome escolhido pelo autor.

A função `var()`, que permite que você possa usar esses valores em outras propriedades.

Aqui está um exemplo:

```
:root {  
  --main-color: #06c;  
}  
  
#foo h1 {  
  color: var(--main-color);  
}
```

`--main-color` é uma propriedade customizada definida pelo desenvolvedor

com um valor de # 06c. Observe que todas as propriedades personalizadas começam com dois travessões.

A função var() recupera e substitui a si mesma pelo valor da propriedade customizada, resultando em color: #06c;Contanto que a propriedade customizada seja definida em algum lugar em sua folha de estilo, ela deve estar disponível para a função var().

Você também pode utilizar as propriedades personalizadas dentro das media queries para ajudar no design responsivo. Um caso de uso pode ser expandir a margem em torno de seus principais elementos de corte à medida que o tamanho da tela aumenta:

```
:root {  
  --gutter: 4px;  
}
```

```
section {  
  margin: var(--gutter);  
}
```

```
@media (min-width: 600px) {  
  :root {  
    --gutter: 16px;  
  }  
}
```

Por que você usaria propriedades personalizadas de CSS nativas?

Você pode usá-los sem a necessidade de um pré-processador.

Eles caem em cascata. Você pode definir uma variável dentro de qualquer seletor para definir ou substituir seu valor atual.

Quando seus valores mudam (por exemplo, media query ou outro estado), o navegador redesenha conforme necessário .

Você pode acessar e manipulá-los em JavaScript.



Material Adicional

Leia: [Utilizando variáveis CSS](#)

Leia: [Medium](#)

MÓDULO D - PUBLICANDO SEU WEBSITE

AULA 1 - REVISÃO DO CONTEÚDO

Esta aula é apenas um breve guia de css para te ajudar naquele momento em que você não lembra exatamente qual propriedade utilizar ou quais valores você pode passar para certa propriedade, entre outras situações.

Então aqui vão algumas referências (em inglês) para você ter sempre na mão quando bater uma dúvida.

*Itens 1 e 2 você pode fazer o download do pdf, o que eu recomendo, já que na página em si não está muito legível

**Item 1 você também pode clicar nas imagens para visualizar em outra aba e em melhor qualidade

1. [Complete CSS Cheat Sheet](#)
2. [CSS Cheat Sheet – A Complete Guide for Beginners and Professionals](#)

3. [Quick and Practical CSS Cheat Sheet](#)



Bônus

[W3Schools CSS Reference](#)

[W3Schools HTML Reference](#)

*Também com comentários sobre tags depreciadas

REFERÊNCIAS HTML/CSS

Nessa aula, separamos algumas das principais referências sobre os principais e mais utilizados assuntos da sprint.

Se precisar relembrar de algum conceito, é só dar um pulo aqui.

HTML Moderno

Tags html:

1. Leia: [Elementos HTML](#)
2. Leia: [HTML elements reference\(em inglês\)](#)
3. Leia: [Seções e estrutura de um documento HTML5](#)

CSS Básico

Introdução:

1. Leia: [Como o CSS é estruturado](#)

Seletores:

1. Leia: [CSS selectors](#)

Display:

1. Leia: [Display \(em inglês\)](#)

Principais displays: block, inline-block, flex, grid e none

Fontes:

1. Leia: [fonts \(em inglês\)](#)

Principais propriedades: font-family, font-size, font-weight, font-style, font-variant, line-height

2. Leia: [Como utilizar o Google Fonts](#)

Unidades de medida:

1. Leia: [CSS Units\(em inglês\)](#)

Dimensões:

1. Leia: [Dimensões - resumo](#)

Columns:

1. Leia: [CSS Multiple Columns\(em inglês\)](#)

Avançando CSS

Box Model:

1. Leia: [O Box Model](#)

Margin e padding:

1. Leia: [CSS margin property\(em inglês\)](#)
2. Leia: [CSS padding property\(em inglês\)](#)

Bordas:

1. Leia: [CSS Borders\(em inglês\)](#)

Avançando Posicionamento CSS

Flexbox:

1. Leia: [Guia completo de flexbox](#)

Responsividade CSS

Responsividade:

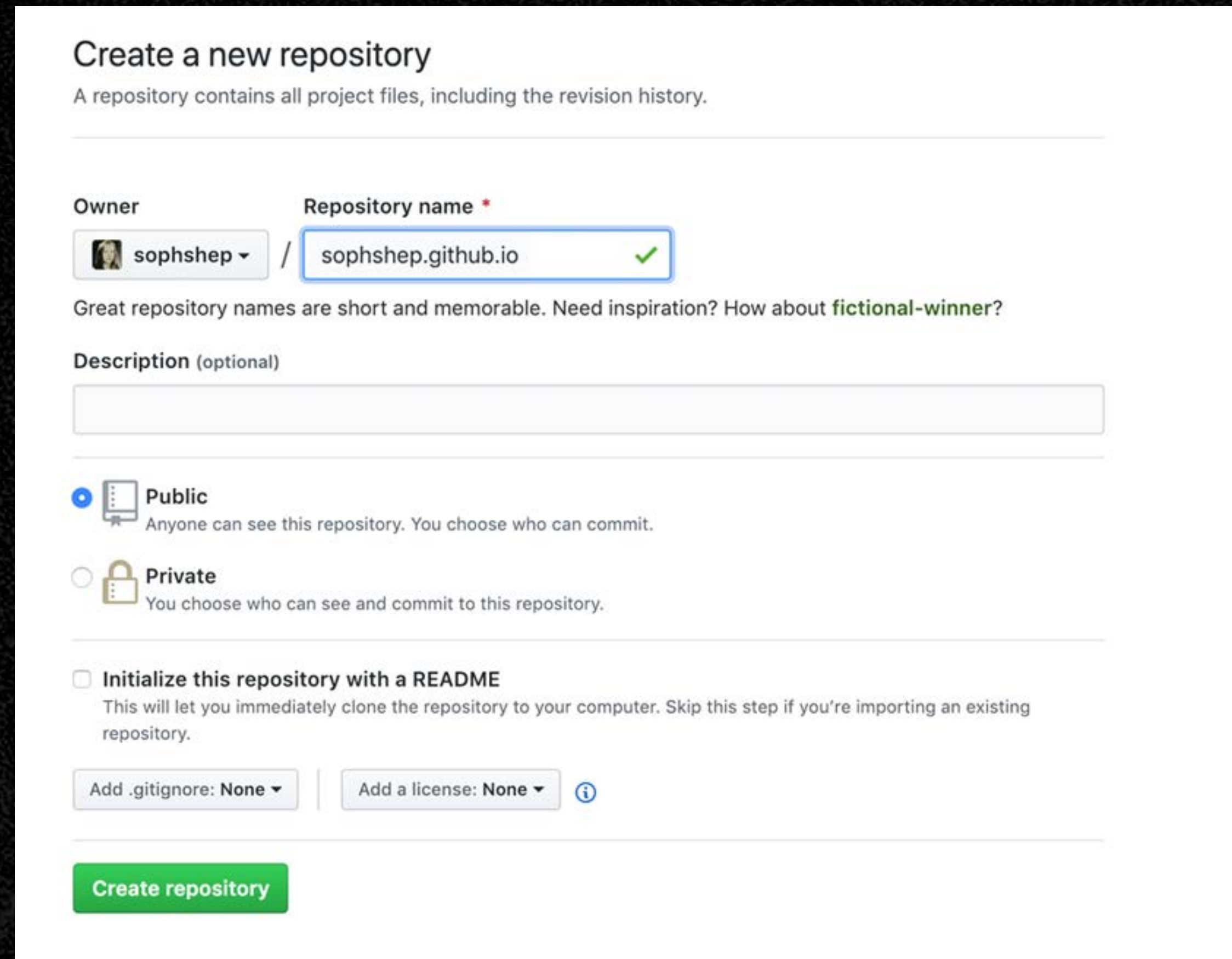
1. Leia: [Design responsivo](#)
2. Leia: [Media queries\(em inglês\)](#)
3. Leia: [CSS media queries\(em inglês\)](#)

AULA 2 - PUBLICANDO SEU SITE

1. CRIE UM REPOSITÓRIO

Vá para o [GitHub](#) e [crie um novo repositório público](#) chamado username .github.io, onde username é o seu nome de usuário (ou nome da organização) no GitHub.

Se a primeira parte do repositório não corresponder exatamente ao seu nome de usuário, não funcionará, então certifique-se de que está correto.

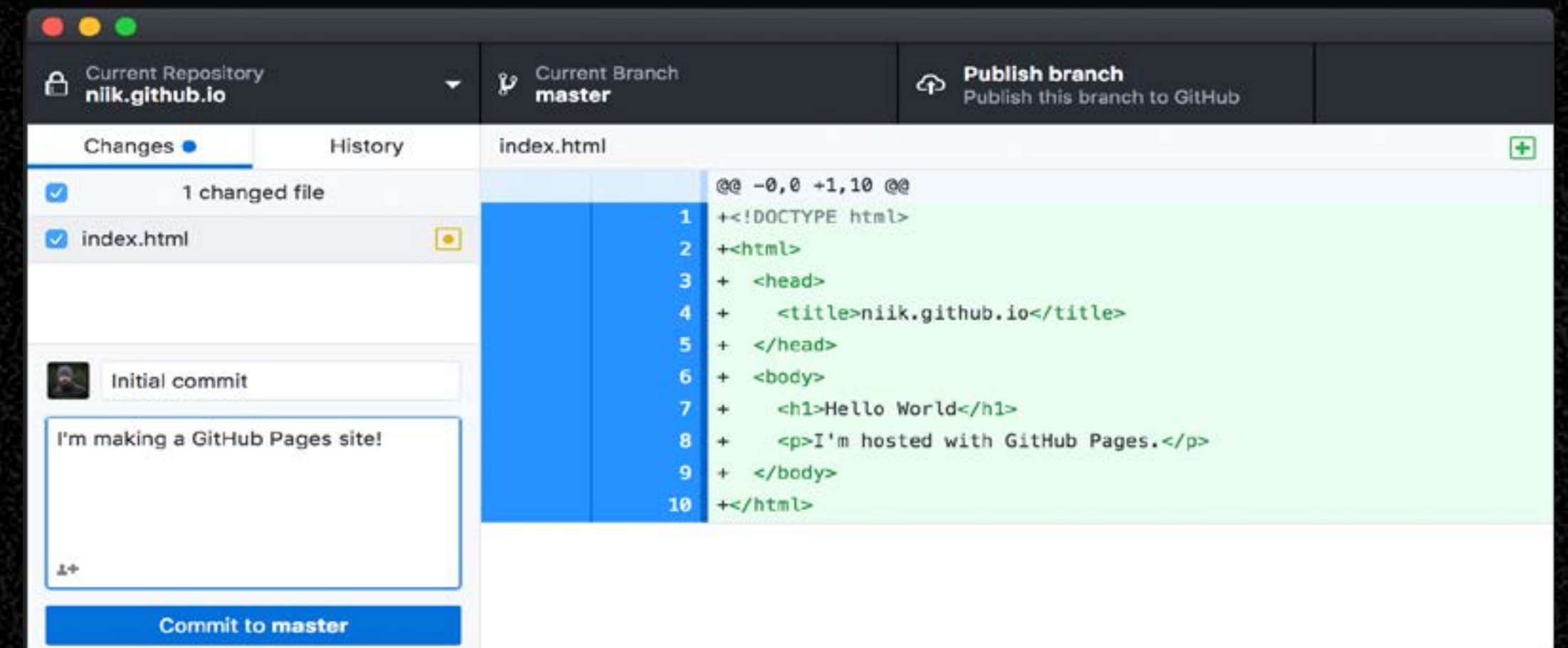


The screenshot shows the 'Create a new repository' page on GitHub. At the top, it says 'Create a new repository' and 'A repository contains all project files, including the revision history.' Below this, there are two main sections: 'Owner' and 'Repository name'. The 'Owner' section shows a dropdown menu with 'sophshep' selected. The 'Repository name' section shows a text input field with 'sophshep.github.io' entered, which is highlighted with a green checkmark. Below these fields, there is a note: 'Great repository names are short and memorable. Need inspiration? How about **fictional-winner**?'. There is also a 'Description (optional)' text input field. Further down, there are two radio button options: 'Public' (selected) and 'Private'. The 'Public' option is described as 'Anyone can see this repository. You choose who can commit.' The 'Private' option is described as 'You choose who can see and commit to this repository.' Below these options, there is a checkbox labeled 'Initialize this repository with a README' with a subtext: 'This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.' At the bottom, there are two dropdown menus: 'Add .gitignore: None' and 'Add a license: None', followed by an information icon. A large green 'Create repository' button is at the very bottom.

2. DOWNLOAD GITHUB DESKTOP

GitHub Desktop é uma ótima maneira de usar Git e GitHub no macOS e Windows.

[Baixe GitHub Desktop](#)



3. CLONE O REPOSITÓRIO

Depois de terminar a instalação, volte para GitHub.com e atualize a página. Clique no botão "Configurar no Desktop". Quando o aplicativo de desktop GitHub for aberto, salve o projeto.

Se o aplicativo não abrir, inicie-o e clone o repositório do aplicativo.



4. CRIE UM ARQUIVO DE ÍNDICE

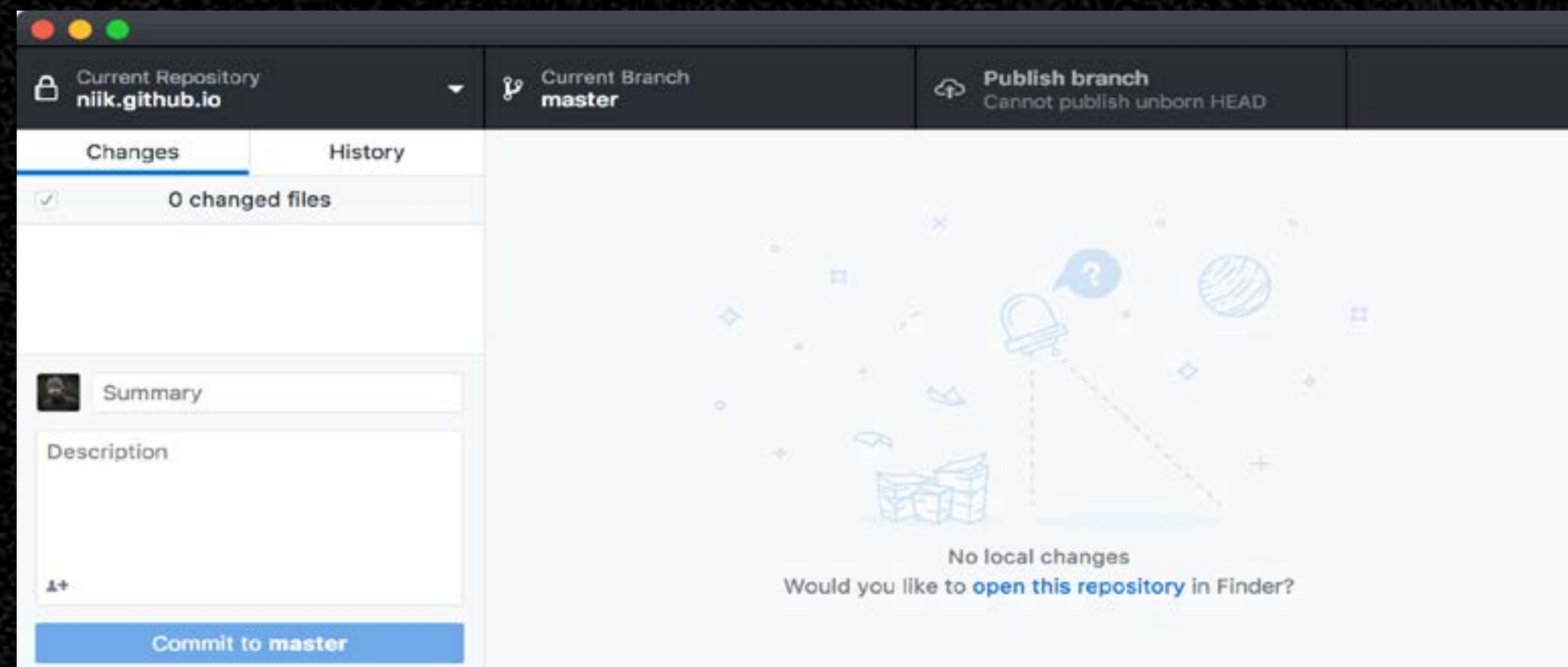
Pegue seu editor de texto favorito e adicione um arquivo index.html ao seu projeto.

5. COMETER E PUBLICAR

Entre no repositório, confirme suas alterações e pressione o botão publicar.

6. ... E PRONTO!

Abra um navegador e acesse <https://username.github.io>.





**Kenzie
Academy**