CSS

Tipografia

Cor

Backgrounds

Cristian Oliveira

Tipografia

São fontes. Coleções de tipografia.

Importadas diretamente para o nosso website.

Através de CSS ou de HTML.

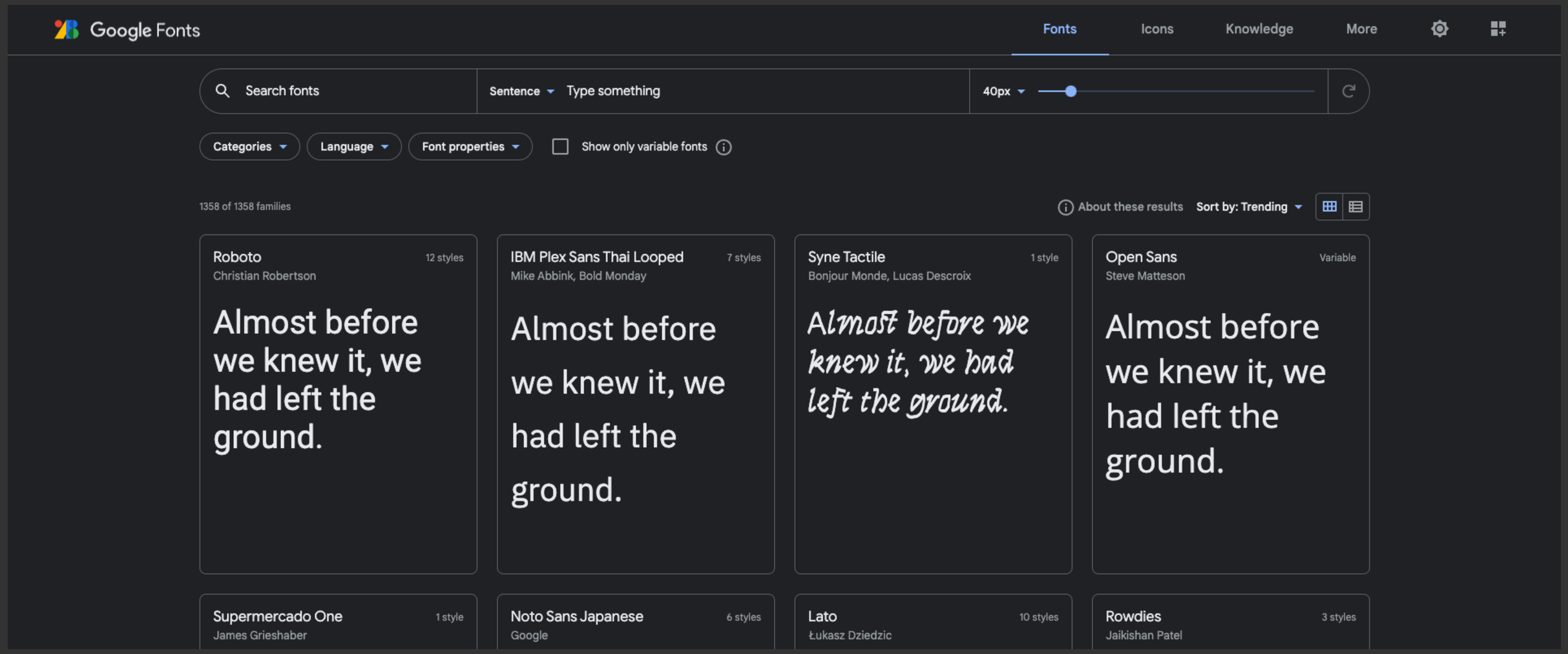
Tipografia

Como qualquer sistema base, existem fontes pré-definidas que podemos utilizar.

No entanto, até certo ponto, não temos que nos contentar com as existentes.

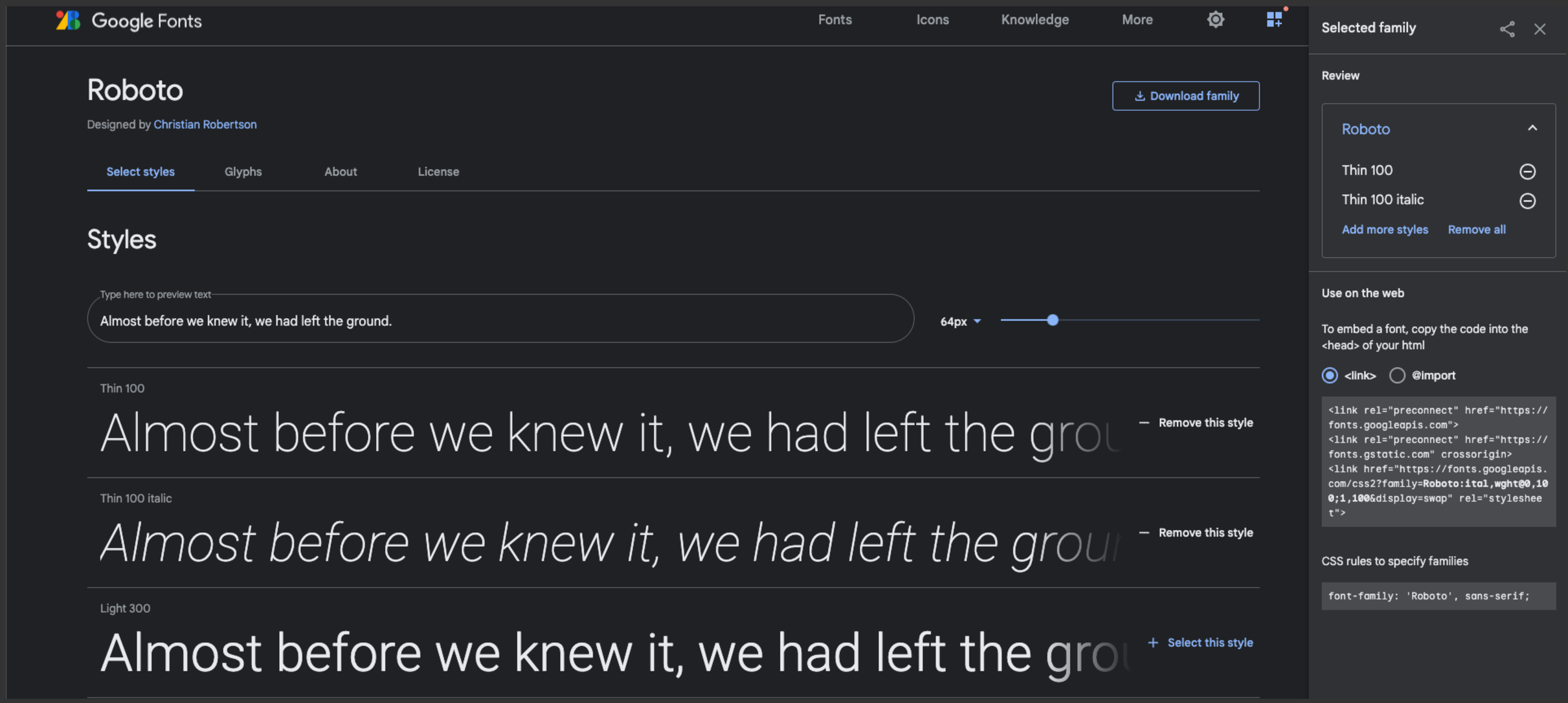
Tipografia

Existem “fornecedores” de fontes que nos podem atribuir fontes mais apelativas que podemos utilizar.



Tipografia

Para utilizar estas fontes, basta utilizar algumas configurações.



Como utilizamos tipografias?

Começamos por adicionar a referência da fonte ao nosso html.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Roboto:ital,
  wght@0,100;1,100&display=swap" rel="stylesheet">
</head>
<body>

</body>
</html>
```

Como utilizamos tipografias?

De seguida, no nosso .css, aplicamos a regras de que o nosso documento usará a fonte escolhida.

```
html {
  font-family: 'Roboto', sans-serif;
}
```

Como utilizamos tipografias?

Outra forma: importação através do ficheiro de .css

Neste caso, não há necessidade de adicionar os elementos no ficheiro .html

```
@import url('https://fonts.googleapis.com/css2?family=Roboto:ital,w-
ght@0,100;1,100&display=swap');

html {
  font-family: 'Roboto', sans-serif;
}
```

Que propriedades de css podemos aplicar aos nossos elementos tipográficos?

color	font-family	font-size	font-style	font-variant	font
letter-spacing	line-height	text-align	text-decoration	text-indent	text-overflow
text-shadow	text-transform	white-space	word-break	word-spacing	

Tudo é relacionado com tudo.
A importância do relativo e do absoluto.

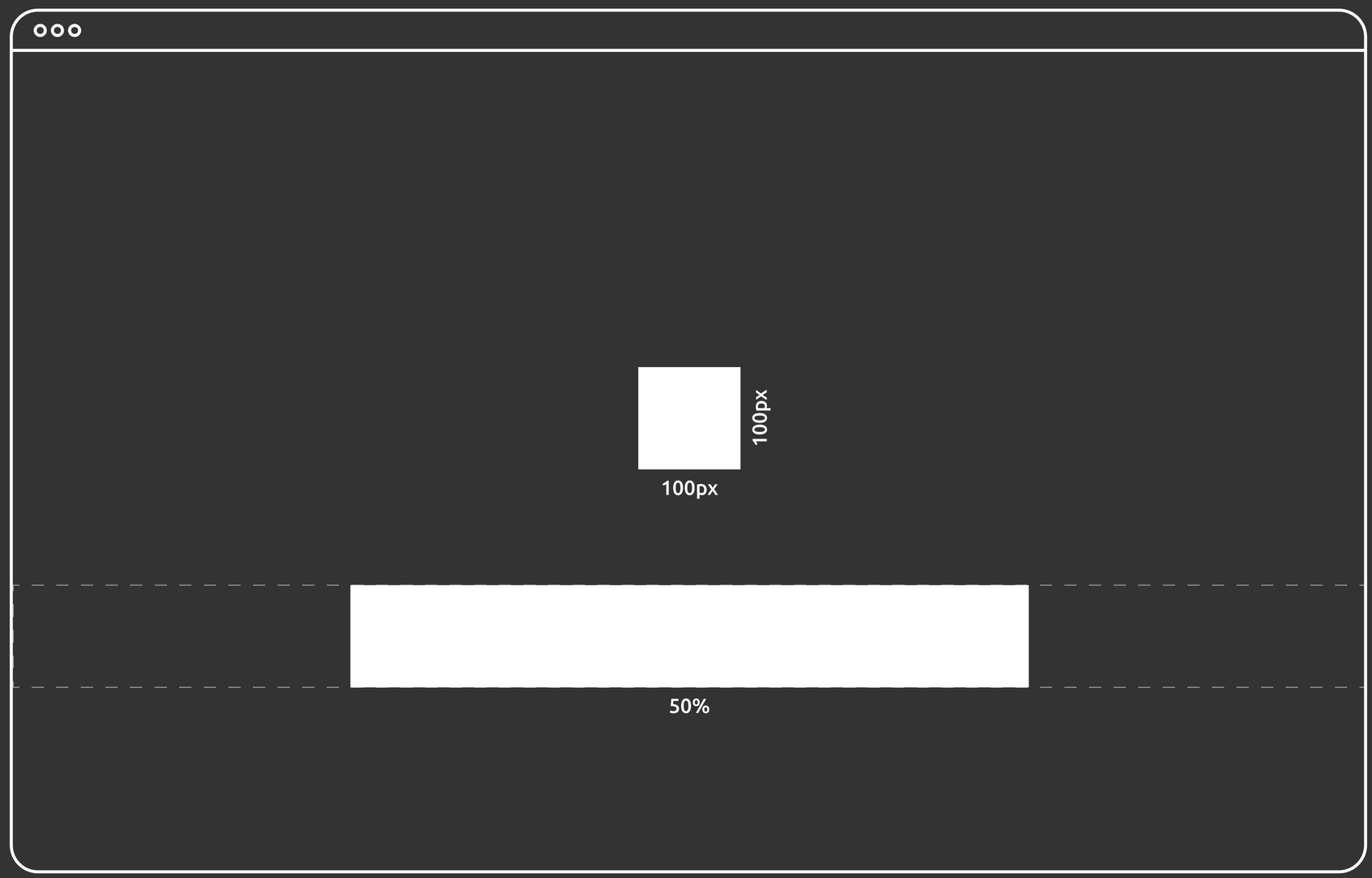
Absoluto

cm	mm	in	pc	pt	px

Relativo

em	ex	ch	rem	lh	vw
vmin	vmax	x			

Relação de medida unitária



100% do viewport - 1% = 14,4px

100vw do viewport - 1vw = 14,4px

1440px de viewport

Unidades absolutas em tipografia

font-size: 16px;

Lorem Ipsum

font-size: 20px;

Lorem Ipsum

Unidades relativas em tipografia

Cor

Cor

nome	hexadecimal	rgb	rgba	hsl	hsla

Cor

red	#FF0000	rgb(255,0,0)	rgb(255,0,0, 1)	hsl(0,100%,50%)	hsl(0,100%,50%,1)

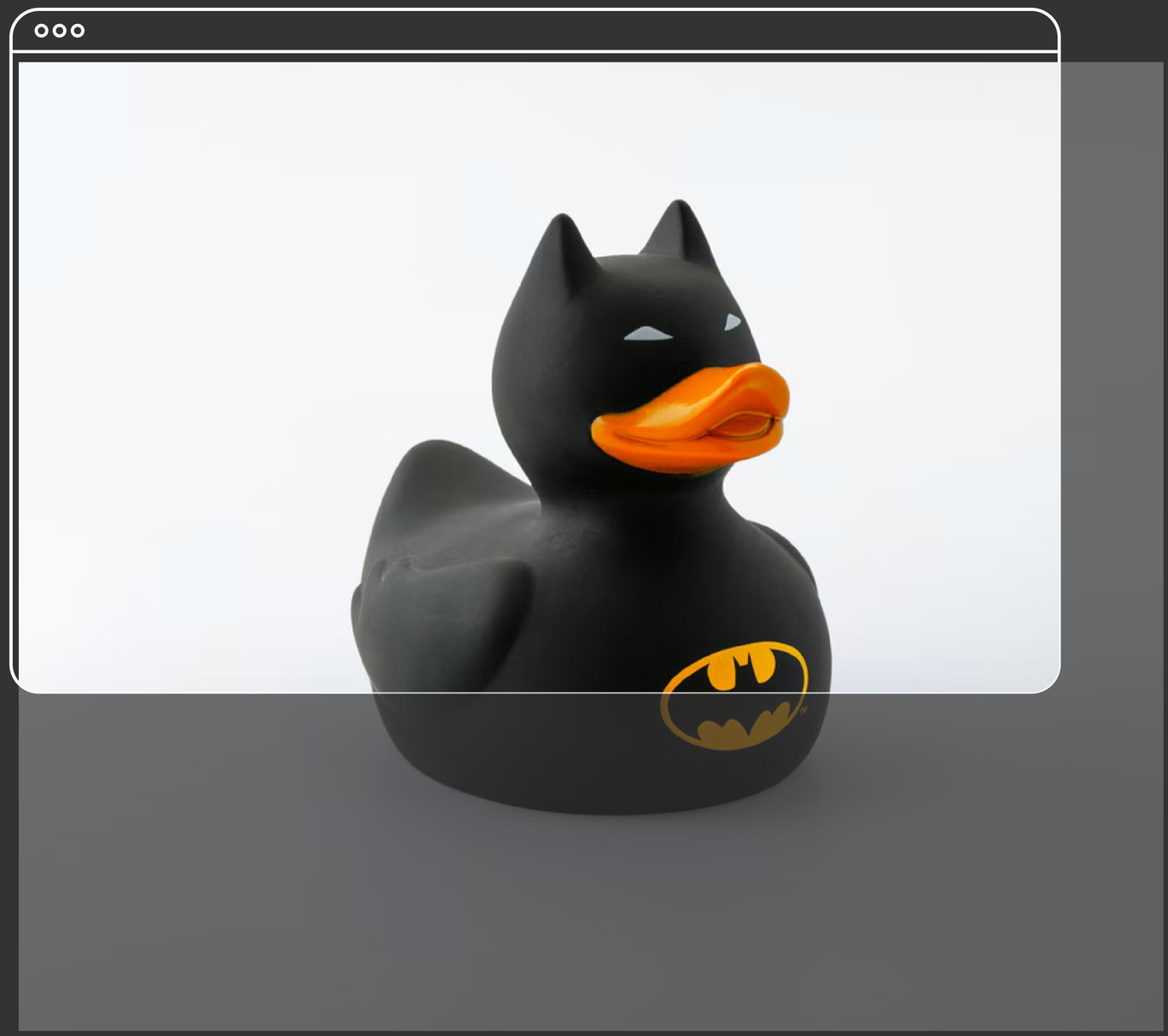
Backgrounds

```
.element {
    background
    background-color
    background-image
    background-position
    background-size
    background-repeat
    background-origin
    background-clip
    background-attachment
}
```

Backgrounds

A propriedade background permite-nos utilizar recursos gráficos e associar por exemplo, imagens ou vídeos ao nosso css.

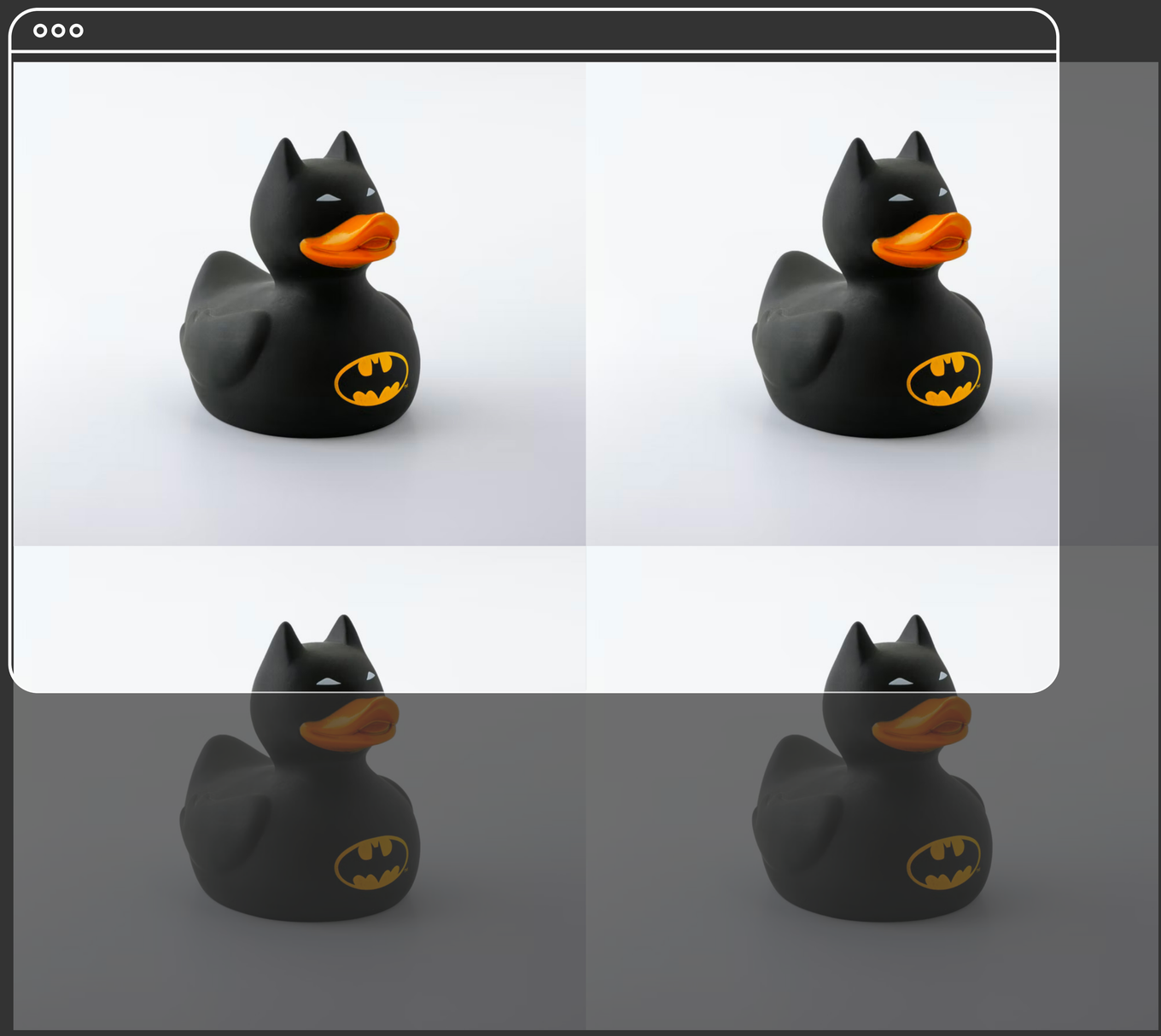
```
.element {
    background-image: url('assets/batman.jpeg')
}
```



Backgrounds

A propriedade background permite-nos utilizar recursos gráficos e associar por exemplo, imagens ou vídeos ao nosso css.

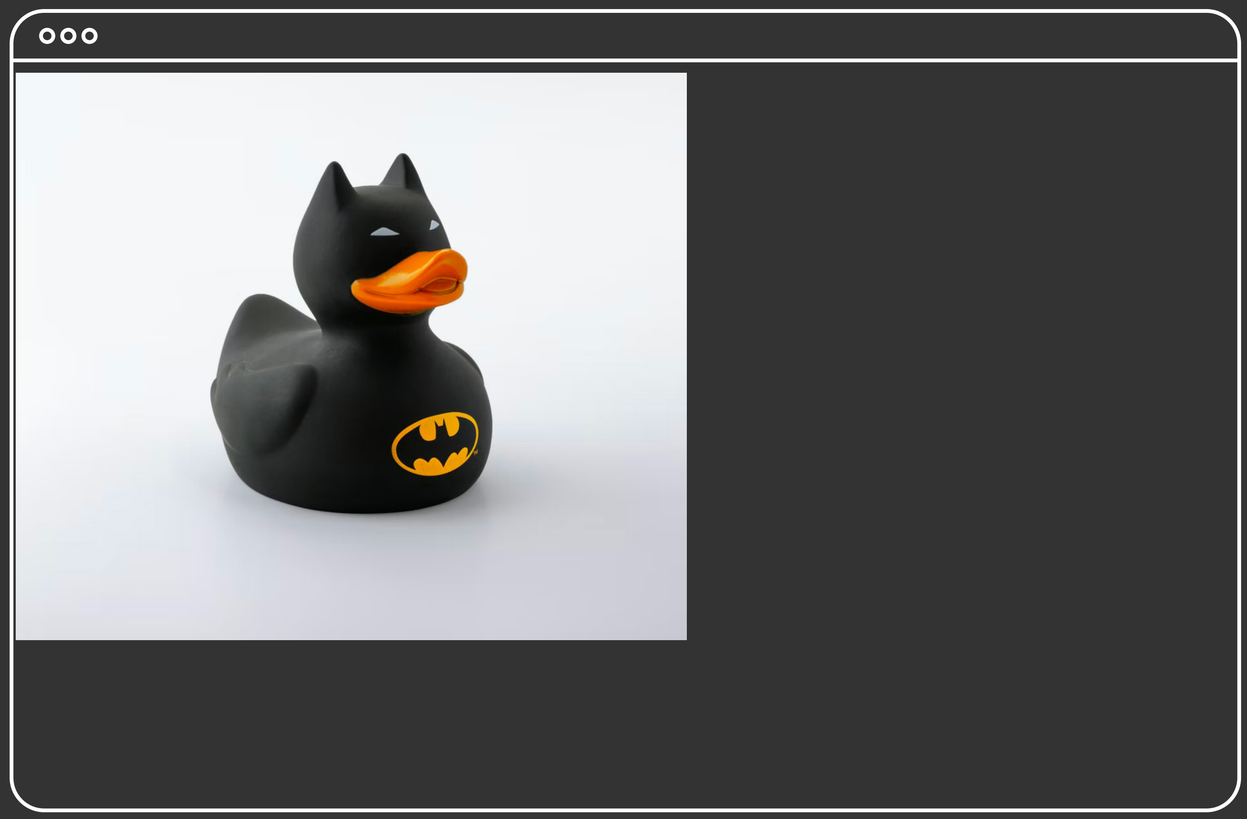
```
.element {
    background-image: url('assets/batman.jpeg');
    background-size: 50%;
}
```



Backgrounds

A propriedade background permite-nos utilizar recursos gráficos e associar por exemplo, imagens ou vídeos ao nosso css.

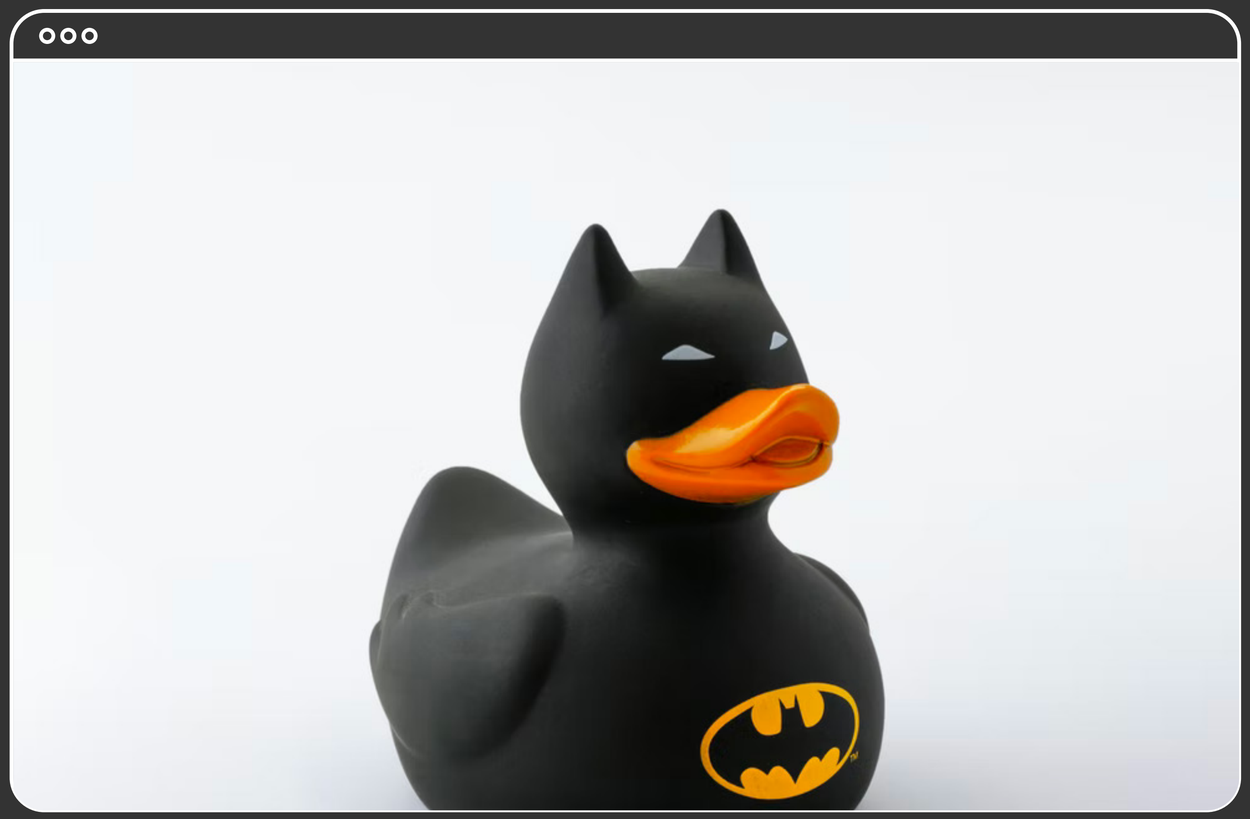
```
.element {
    background-image: url('assets/batman.jpeg');
    background-size: 50%;
    background-repeat: no-repeat;
}
```



Backgrounds

A propriedade background permite-nos utilizar recursos gráficos e associar por exemplo, imagens ou vídeos ao nosso css.

```
.element {
    background-image: url('assets/batman.jpeg');
    background-size: cover;
    background-repeat: no-repeat;
}
```



Backgrounds

A propriedade background permite-nos utilizar recursos gráficos e associar por exemplo, imagens ou vídeos ao nosso css.

```
.element {
    background-image: url('assets/batman.jpeg');
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;
}
```

Backgrounds

A propriedade background permite-nos utilizar recursos gráficos e associar por exemplo, imagens ou vídeos ao nosso css.

```
.element {
    background-color: yellow;
}
```

CSS

Box Model

Box Sizing

Overflows

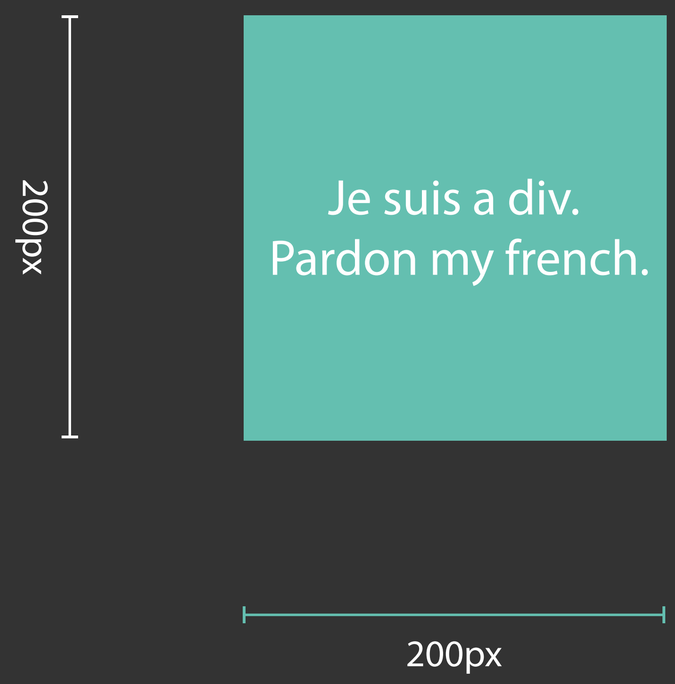
A anatomia dos elementos

Já sabemos que a referência geografica do nosso elemento começa no HTML quando definimos o elemento que estamos a criar. No entanto, existem propriedades no css que nos informam que o elemento tem outro tipo de fronteiras geográficas que nos podem ser úteis. A isto chamamos de box-model

Box Model

No princípio, e quase sempre depois, todos os elementos são rectângulos.

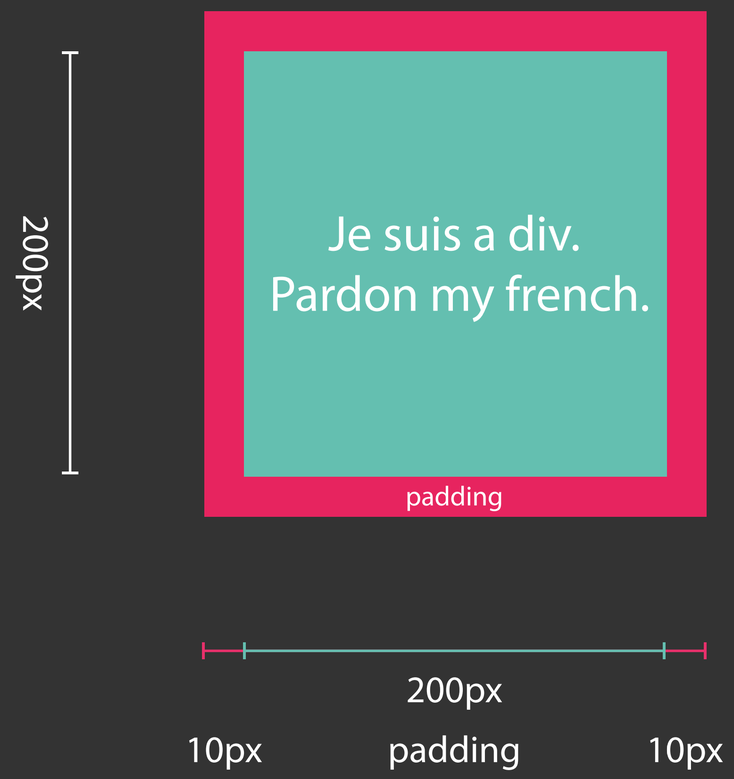
Começa pelos seus limites : width e height



Box Model

De seguida, entramos no domínio do padding. Uma margem interna entra o elemento e a sua border.

padding: 10px;



Box Model

Agora, é a altura da border, o contorno associado ao nosso elemento.

border: 5px;

200px

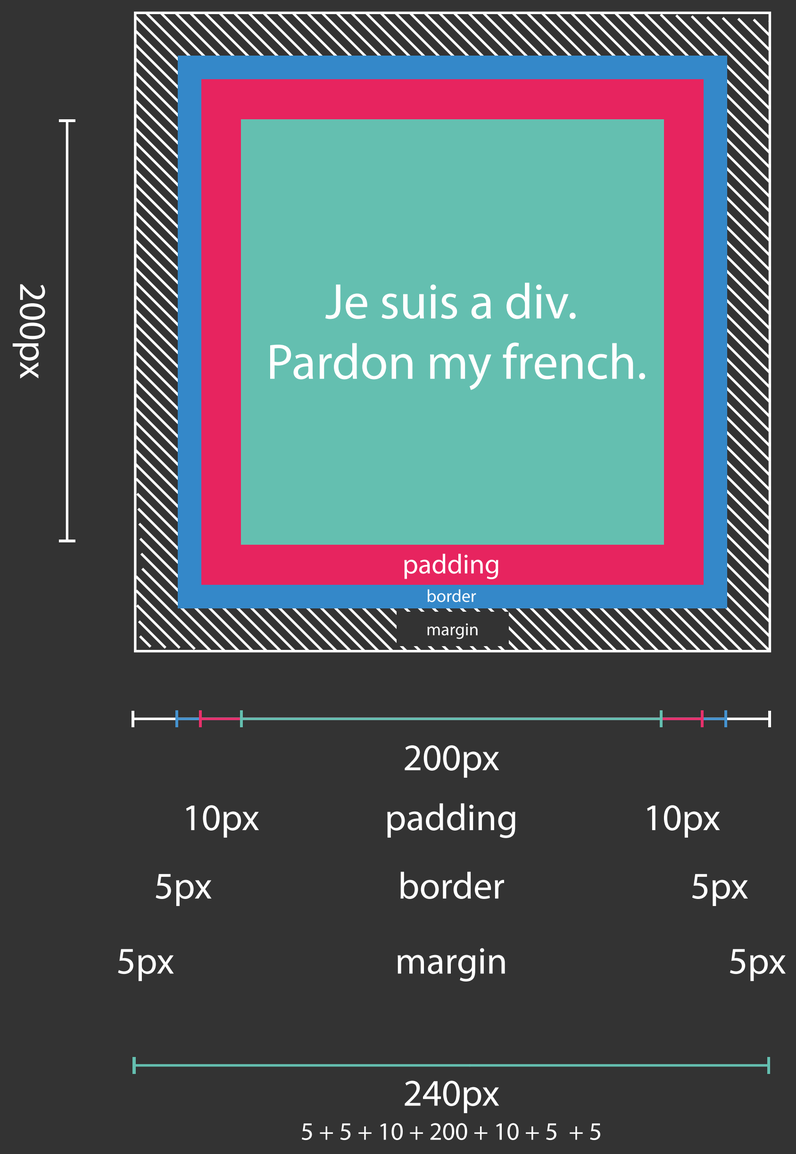
10px padding 10px

5px border 5px

Box Model

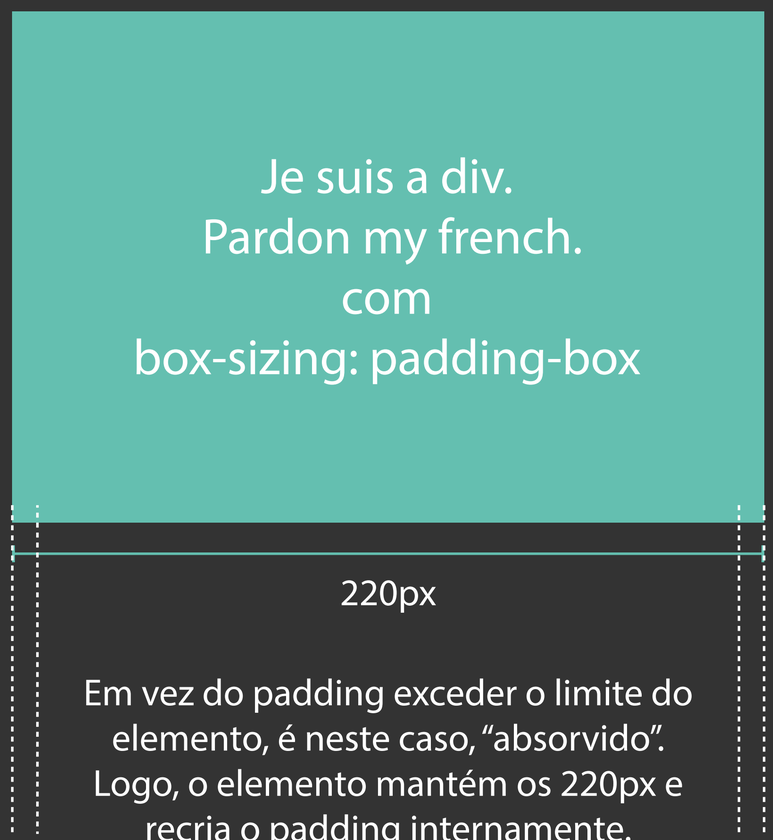
Por fim, a margem, a característica que cria uma margem de segurança entre o elemento e os siblings.

margin: 10px;



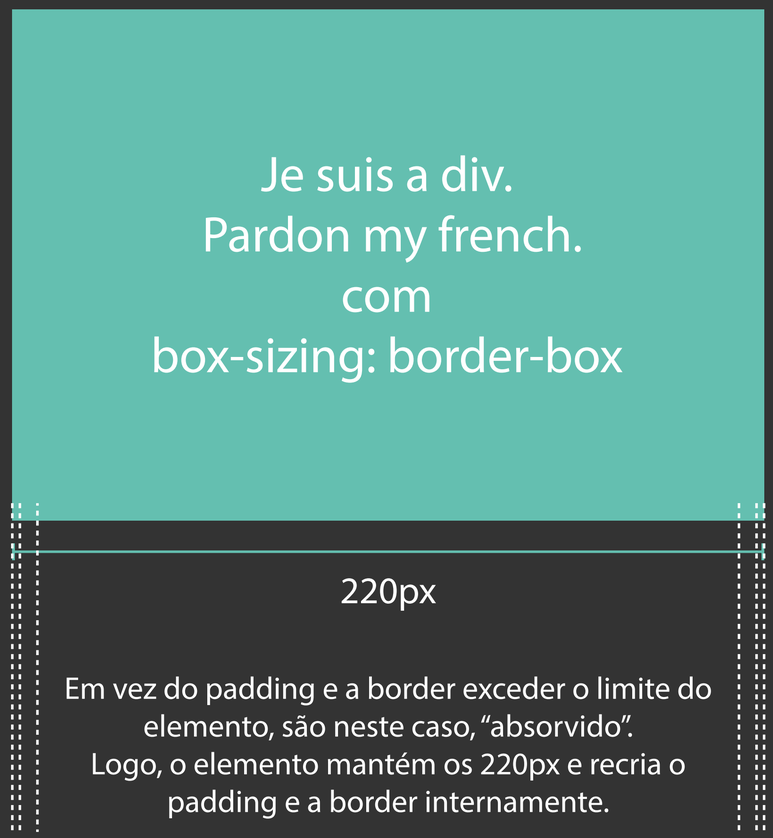
Box Sizing

O box sizing ajuda-nos a delimitar o nosso elemento. Com esta propriedade podemos “absorver” o espaço ocupado pelo padding ou a border



Box Sizing

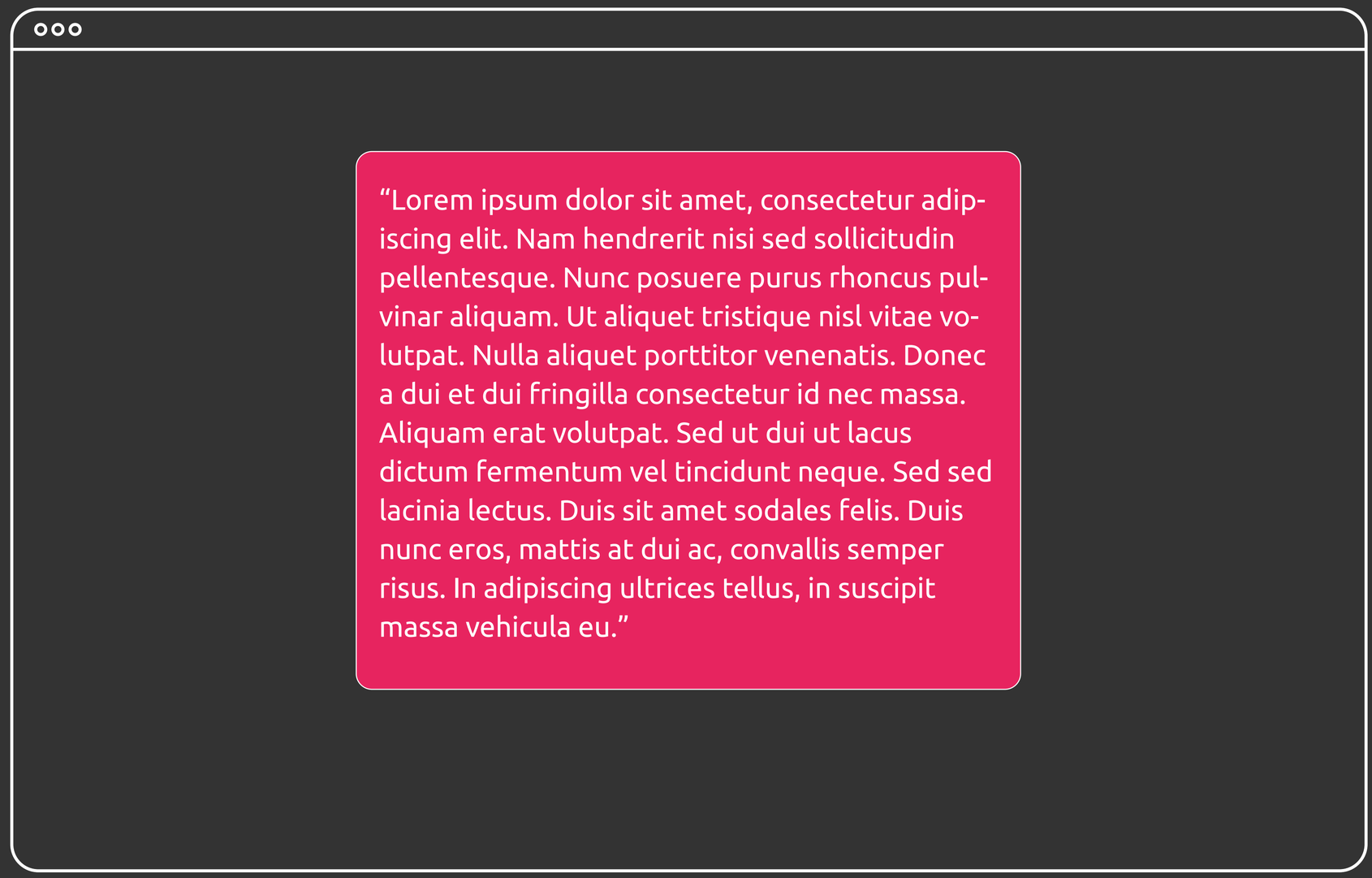
O box sizing ajuda-nos a delimitar o nosso elemento. Com esta propriedade podemos “absorver” o espaço ocupado pelo padding ou a border



A propriedade box-sizing é capaz de absover os valores do padding e a border para dentro do limite do elemento. Mas nunca é capaz de absorver a margin.

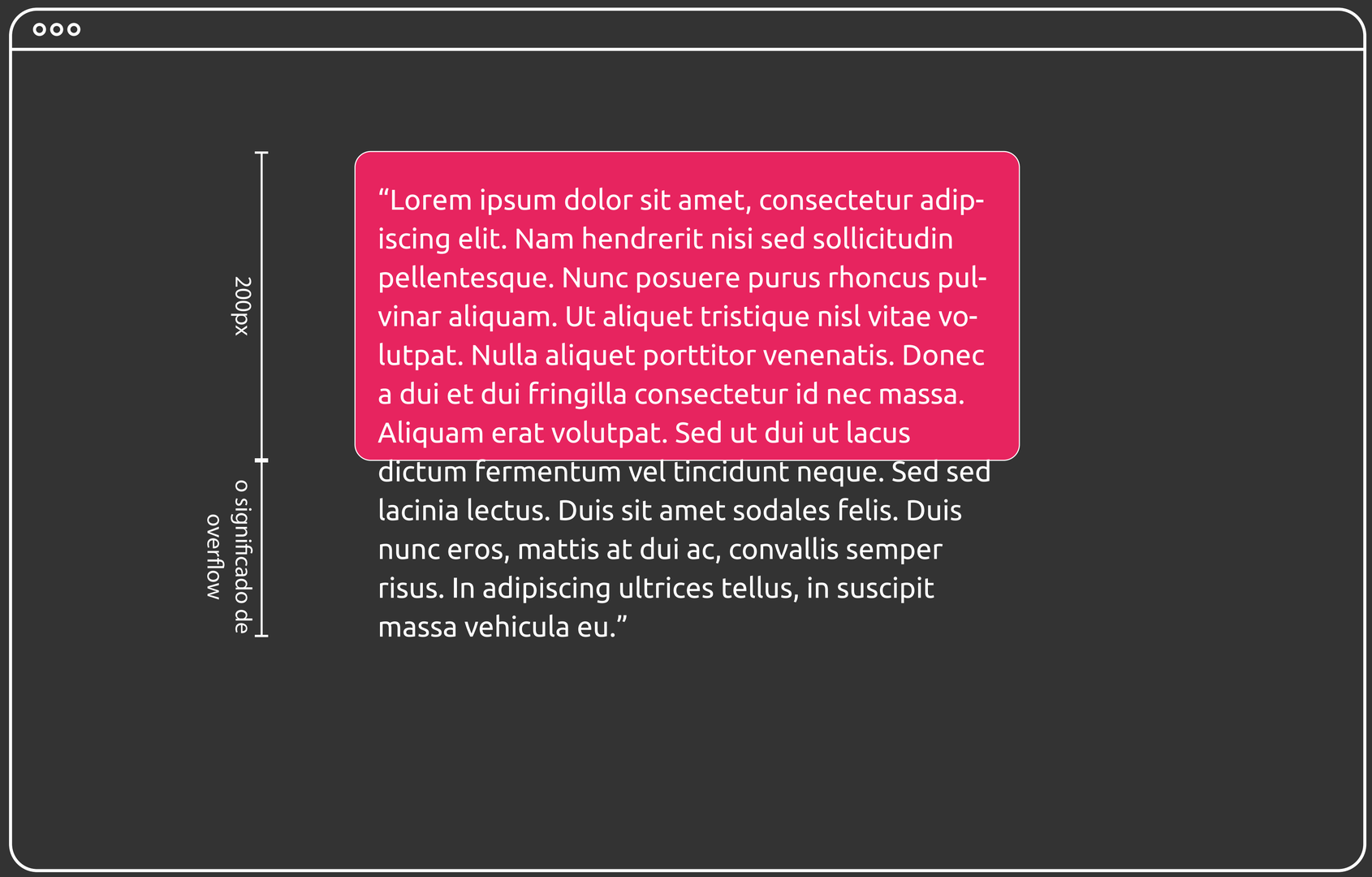
Overflow

É a propriedade que define quando o contéudo do elemento deve ser cortado.



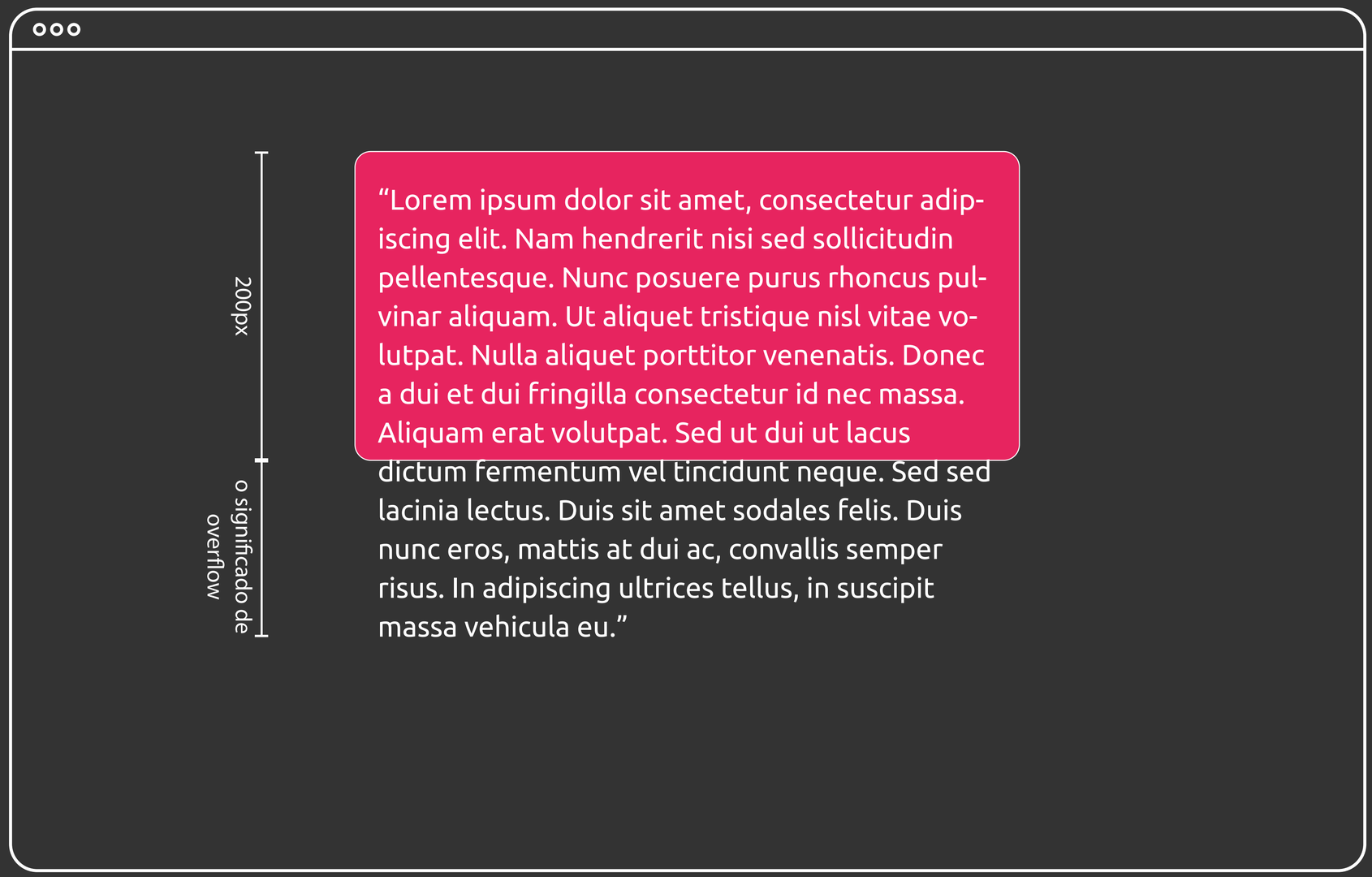
Overflow

É a propriedade que define quando o contéudo do elemento deve ser cortado.



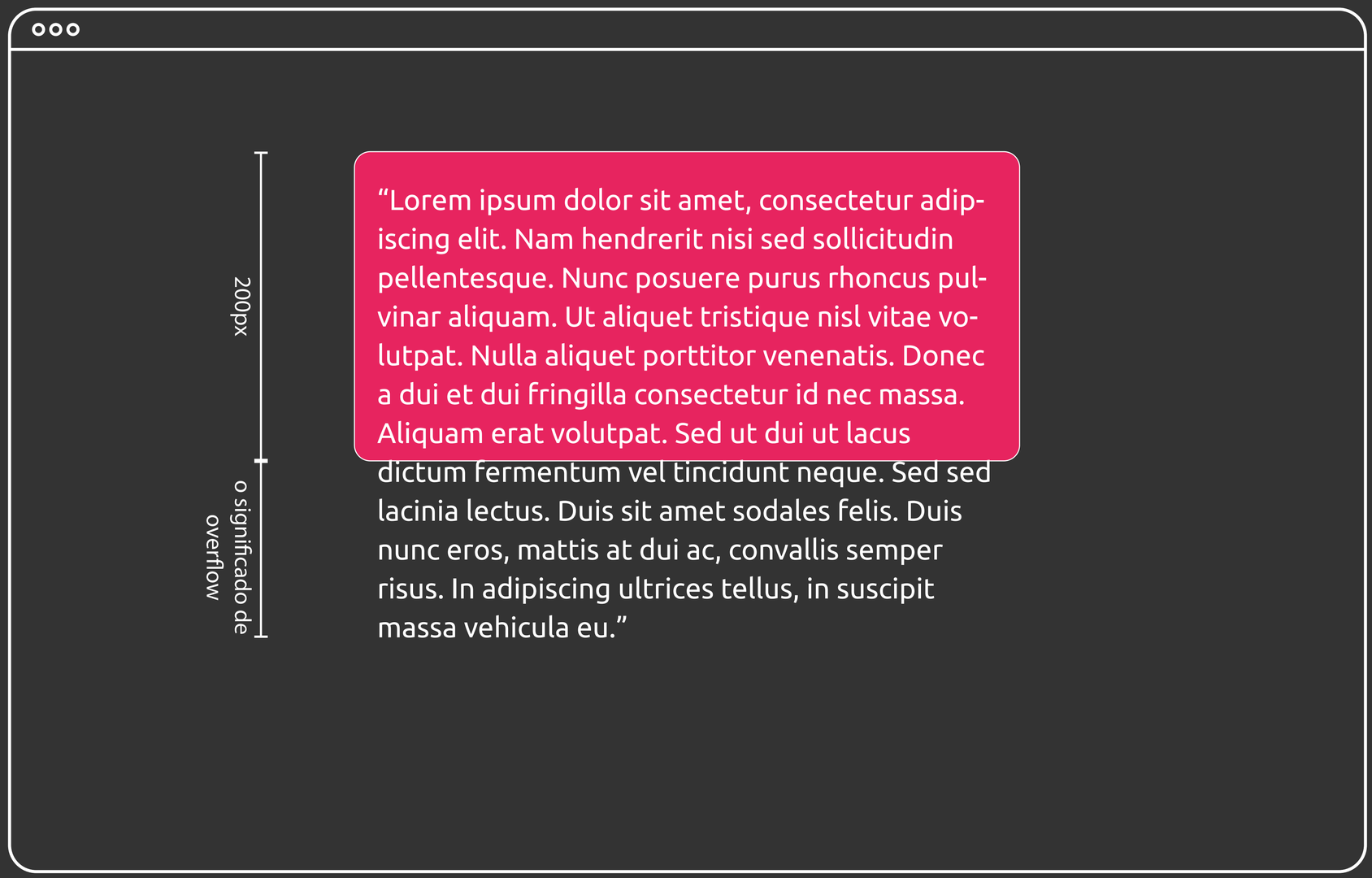
Overflow

É a propriedade que define quando o contéudo do elemento deve ser cortado.



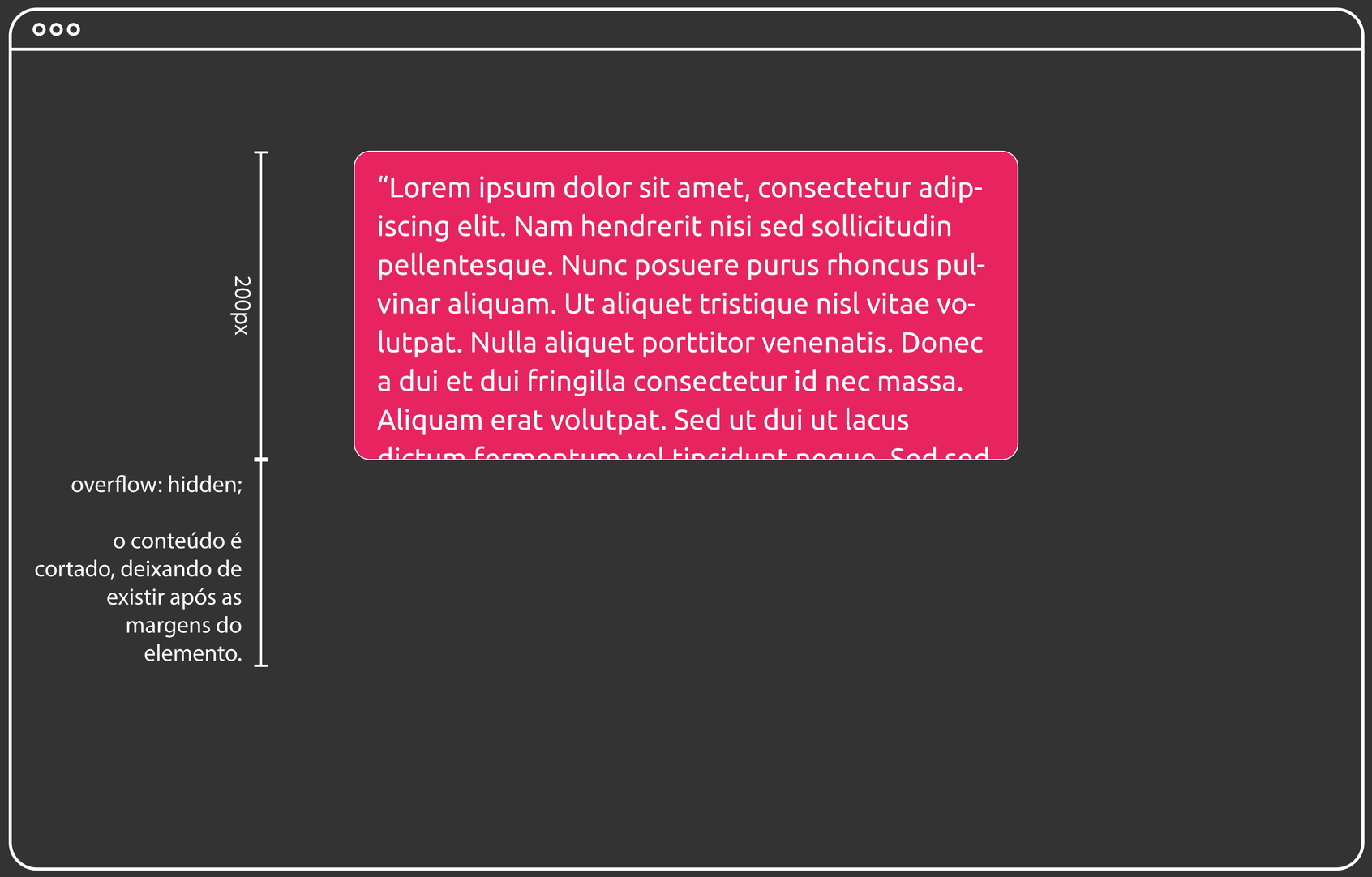
Overflow

É a propriedade que define quando o contéudo do elemento deve ser cortado.



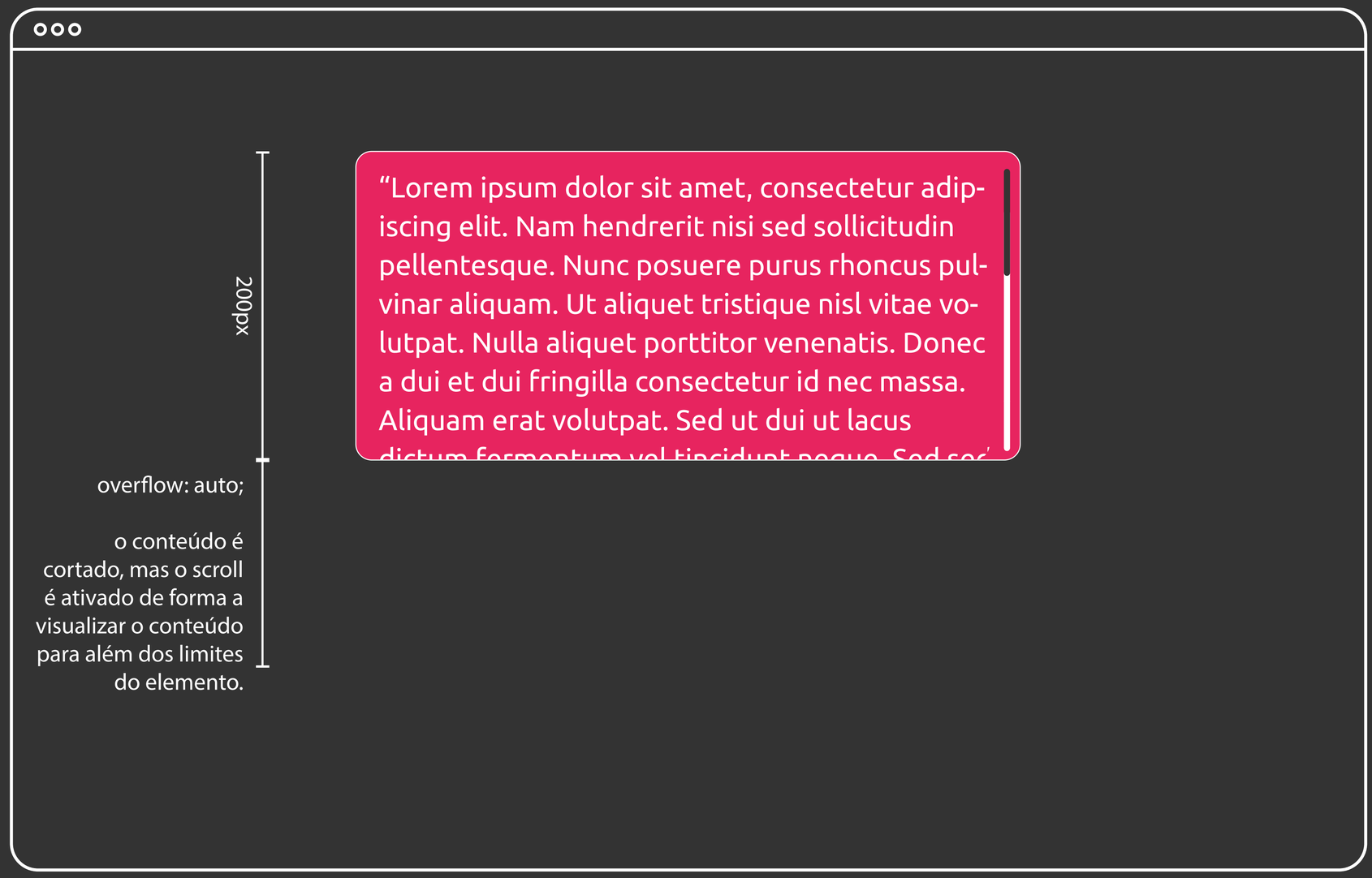
Overflow: hidden

É a propriedade que define quando o contéudo do elemento deve ser cortado.



Overflow: auto

É a propriedade que define quando o contéudo do elemento deve ser cortado.



CSS

Display
Visibility
Position
z-index

Display

Existem propriedades que já nascem com os nossos elementos.

São propriedades que são a natureza do próprio elemento.

Display

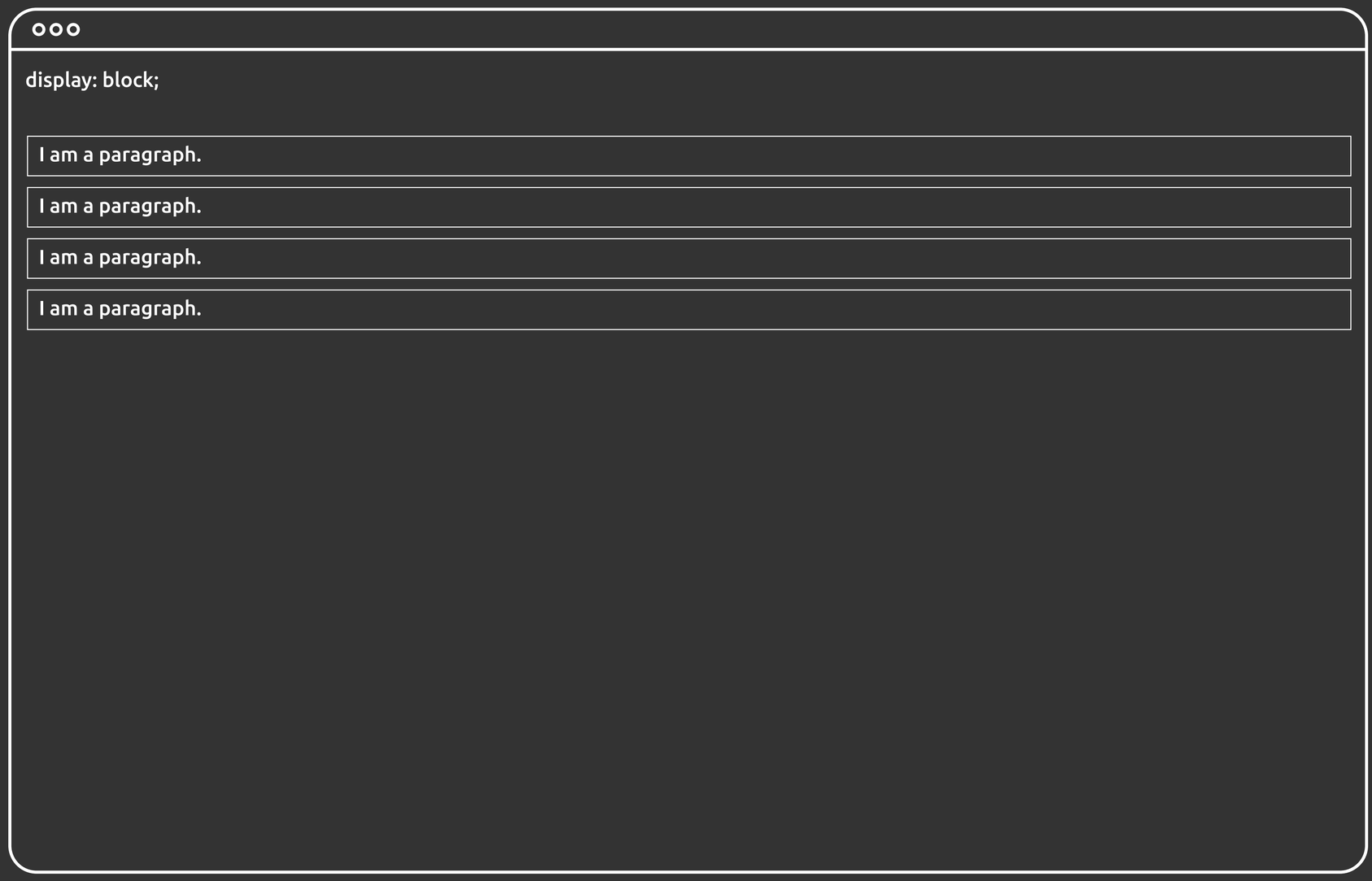
A propriedade display retrata a forma como o elemento se comporta no seu espaço, em relação aos restantes elementos.

inline	block	inline-block	initial	inherit	none

Display: block;

Retrata o elemento como um block.

Ocupa a totalidade da sua largura.



Display: inline;

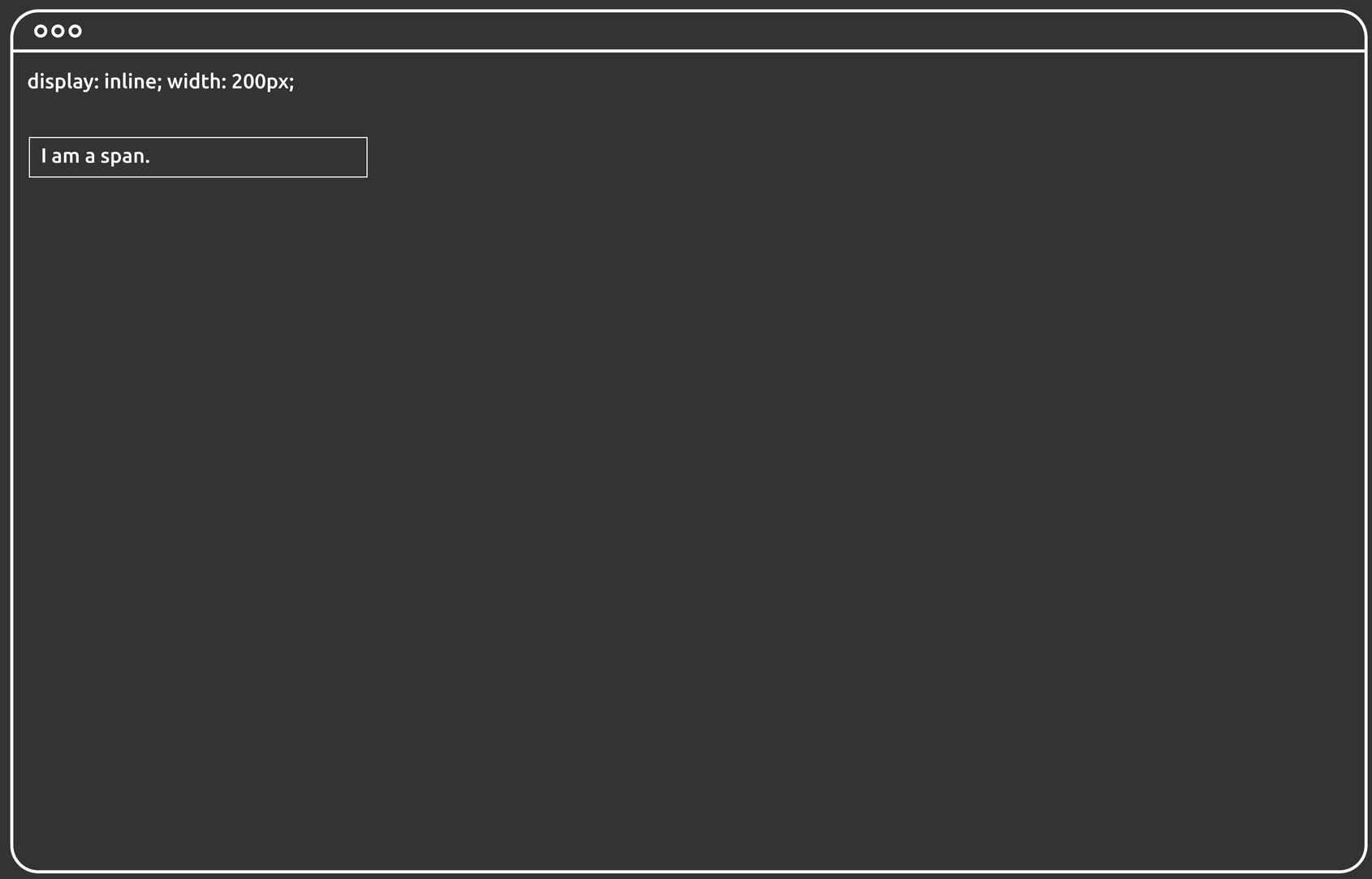
Ocupa o espaço que precisa.

Larguras e Alturas não terão efeito.

Display: inline-block;

Comporta-se como um inline.

Mas aqui sim, a largura e altura já produzem efeito.



Display: initial;

Formata o elemento para o seu valor natural, inicial.

Display: inherit;

O filho herda o valor do pai.

Display: none;

O elemento é completamente removido da DOM

Display

Existem ainda outras propriedades com propósitos diferentes.

inline	block	inline-block	initial	inherit	none
flex	grid	inline-flex	inline-grid	inline-table	list-item
table	run-in	table-caption	table-cell	table-column	table-row

Visibility

A propriedade visibility parece um display:none, mas não o é.

A diferença aqui, é que o visibilty: hidden faz o elemento desaparecer, mas o seu espaço continua a existir

Position

Tudo é relacionado com alguma coisa.

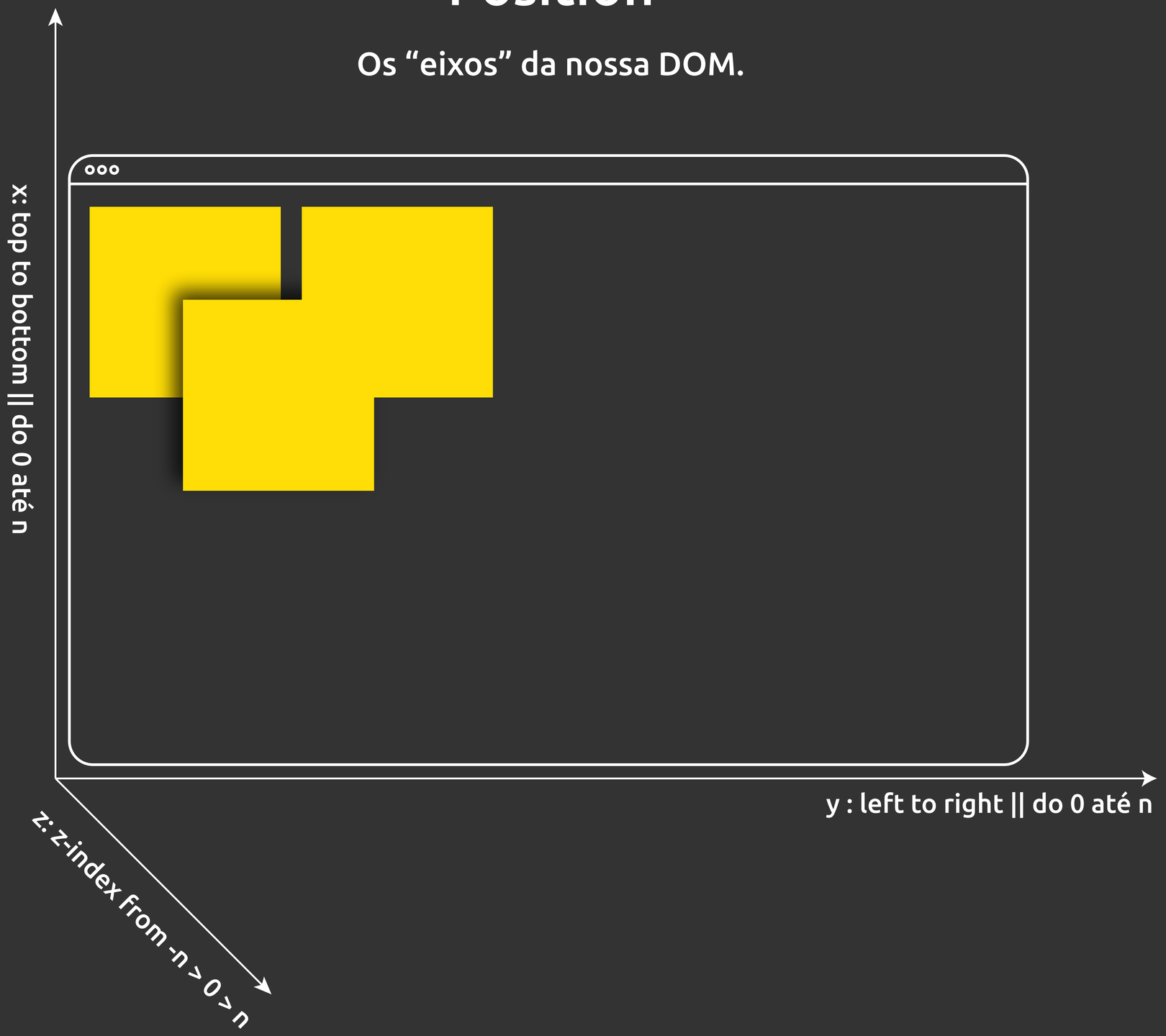
Position

A propriedade position garante o comportamento de um elemento em relação aos restantes elementos da página.

Permite-nos tratar cada elemento com características únicas em relação aos seus outros elementos da página.

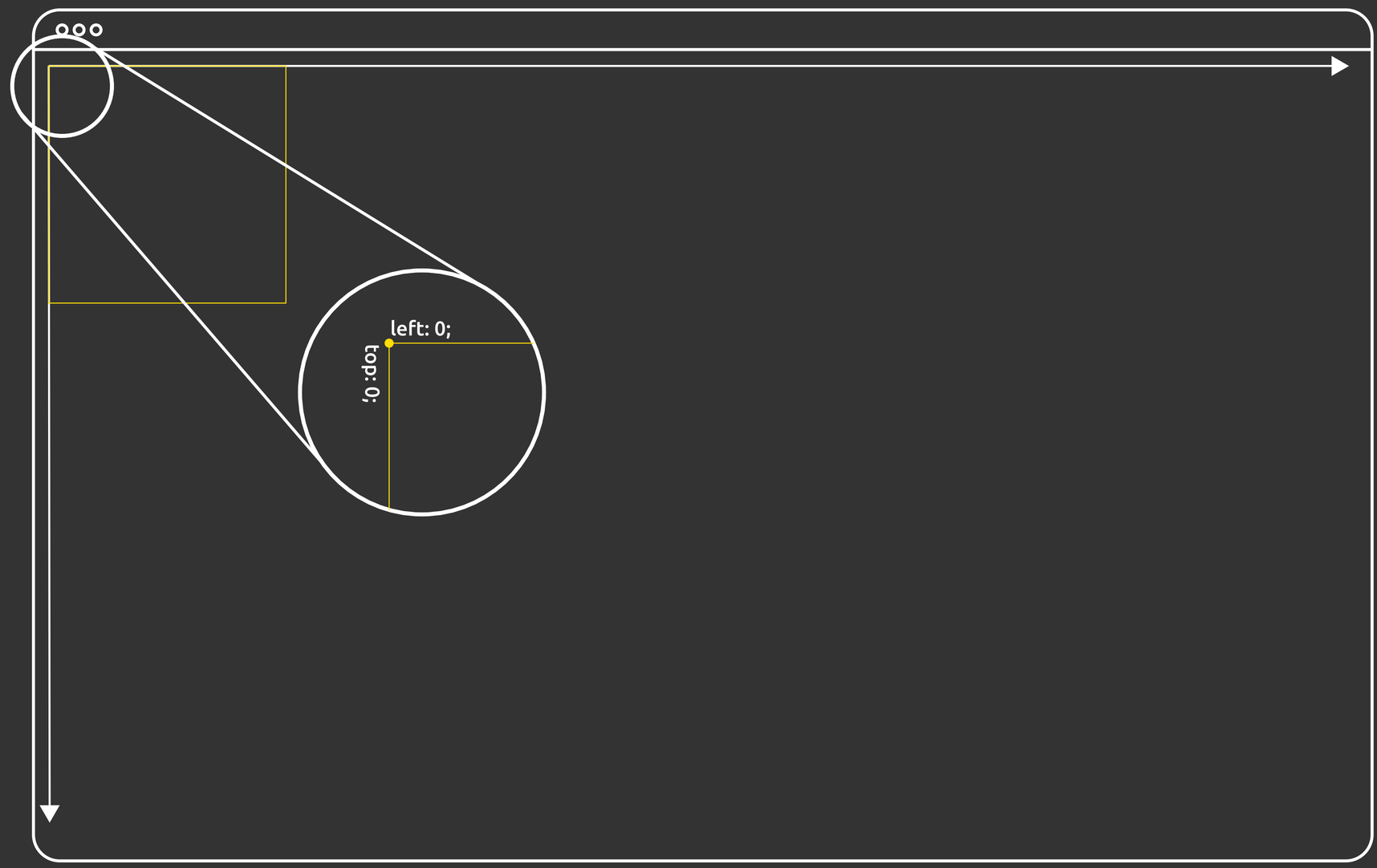
Position

Os “eixos” da nossa DOM.



Position

O ponto de origem do eixo é por default o canto superior esquerdo.



Position

A propriedade position refere o posicionamento do elemento no eixo x e y.

Os elementos são static por default. Existem também os seguintes valores:

static	relative	fixed	absolute	sticky	

Position: static

Um elemento static não tem posicionamento específico.
É posicionado de acordo com o flow normal dos elementos e não é afetado pelas propriedades top, right, bottom ou left.

Position: relative

Um elemento relative é posicionamente relativamente a algo.

A sua posição inicial pode ser afetada pelas propriedades top, right, bottom ou left.

Position: absolute

Um elemento absolute é posicionamente relativamente ao seu ancestral relativo mais próximo.

A sua posição inicial é afetada pelas propriedades top, right, bottom ou left.

E não tem irmãos posicionamentos relativamente.

Position: fixed

Muito similar ao absolute. No entanto, o fixed contém a particularidade de ser fixo ao viewport, não é afetado pelo scroll. É também afetado pelas propriedades top, right, bottom e left.

Position: fixed

O sticky contém a particularidade de ser relativo aos seus adjacentes mas especialmente as suas coordenadas na página. No scroll, quando este elemento chega à coordenada desejada, cola. Torna-se fixed. Aqui, o top, right, bottom e left não o posicionado, mas informam onde será o seu destino.

z-index

O z-index é responsável pelo positionamento do elemento no eixo z.

O positionamento dos elementos é feito como se fosse uma camada de layers sobrepostos dando o efeito de “trazer elementos para a frente ou para trás de outros elementos”

Só funciona com position: absolute, position: relative, position: fixed, ou position: sticky