

```
* {  
  margin: 0;  
  padding: 0;  
  box-sizing: border-box;  
  font-family: "Poppins", sans-serif;  
}
```

Esse bloquinho de CSS é bem comum e **super importante** no início de qualquer projeto. Ele ajuda a garantir que os estilos fiquem **consistentes** em todos os navegadores.

🎯 Seletor universal *

```
* {
```

- O * é o **seletor universal**.
- Ele aplica as regras CSS a **todos os elementos da página**: body, h1, div, input, button, etc.
- Muito útil para aplicar resets ou estilos globais.

📏 margin: 0;

```
margin: 0;
```

- Remove **todas as margens externas** padrão que os navegadores colocam.
- Por exemplo, o <body> normalmente tem uma margem automática de 8px. Isso limpa isso.

📏 padding: 0;

```
padding: 0;
```

- Remove **todos os espaçamentos internos** (padding) padrões dos elementos.

box-sizing: border-box;

`box-sizing: border-box;`

- Essa propriedade muda a forma como o navegador calcula as **larguras e alturas dos elementos**.
- Com border-box, a **largura total inclui padding e borda**, o que facilita muito no layout.

Exemplo prático:

```
width: 200px;  
padding: 20px;
```

- Com `box-sizing: content-box` (padrão antigo), o elemento ficaria com 240px de largura ($200 + 2 \times 20$).
- Com border-box, o **elemento continua com 200px no total**. Bem mais fácil de controlar.

font-family: "Poppins", sans-serif;

`font-family: "Poppins", sans-serif;`

- Define a fonte padrão da página como **Poppins** (moderna, limpa e estilosa).
- Se o navegador não tiver a fonte Poppins instalada, ele usa qualquer outra fonte `sans-serif` (sem serifa) como fallback.

Obs: Para usar Poppins corretamente, você precisa importar ela do Google Fonts ou do seu projeto.

Resumo geral:

Esse código:

- **Remove os estilos padrões dos navegadores** (margem, padding).
- **Simplifica os cálculos de tamanho com border-box**.

- **Aplica uma fonte moderna (Poppins) como padrão.**

É tipo um **reset inteligente** com um toque de estilo! 💖 ✨

```

.logo-mario {
  height: 200px;
}

button {
  color: rgb(7, 0, 0);
  background: rgb(12, 190, 146);
  border-radius: 3px;
  border: none;
  width: 200px;
  height: 50px;
  cursor: pointer;
  text-transform: uppercase;
  font-weight: bold;
  margin-top: 30px;
}

button:hover {
  background-color: rgb(12, 190, 27);
  color: white;
  font-size: 15px;
  transition: 0.5s ease-in-out;
}

.imagem-mario-luigi {
  height: 350px;
}

```

Esse trecho de CSS define estilos para elementos como **uma logo, um botão e uma imagem do Mario & Luigi**.

.logo-mario

```

.logo-mario {
  height: 200px;
}

```

- Seleciona um elemento com a classe logo-mario.
- Define que a altura da logo será **200 pixels**.
- Largura será ajustada automaticamente conforme a proporção da imagem (se for).

○ button

```
button {  
  color: rgb(7, 0, 0);  
  background: rgb(12, 190, 146);  
  border-radius: 3px;  
  border: none;  
  width: 200px;  
  height: 50px;  
  cursor: pointer;  
  text-transform: uppercase;  
  font-weight: bold;  
  margin-top: 30px;  
}
```

Esse bloco estiliza **todos os botões da página**:

Propriedade	O que faz
color	Cor do texto do botão (um tom de preto bem escuro)
background	Cor de fundo: verde água (verde-azulado)
border-radius: 3px	Cantos arredondados levemente
border: none	Remove a borda padrão do botão
width: 200px	Largura de 200 pixels
height: 50px	Altura de 50 pixels
cursor: pointer	Mostra a "mãozinha" ao passar o mouse
text-transform: uppercase	Transforma o texto em letras maiúsculas
font-weight: bold	Deixa o texto em negrito
margin-top: 30px	Dá um espaço de 30px acima do botão

👆 button:hover

```
button:hover {  
  background-color: rgb(12, 190, 27);  
  color: white;  
  font-size: 15px;  
  transition: 0.5s ease-in-out;
```

```
}
```

Esse bloco ativa **quando o mouse passa por cima do botão (hover)**:

Propriedade	O que muda no hover
background-color	Muda para um verde mais forte
color	Texto muda para branco
font-size: 15px	Aumenta um pouco o tamanho da fonte
transition	Cria uma transição suave de 0.5s com efeito de entrada/saída

.imagem-mario-luigi

```
.imagem-mario-luigi {  
  height: 350px;  
}
```

- Seleciona a imagem com a classe `imagem-mario-luigi`.
- Define a altura dela como **350 pixels**.

Provavelmente usada para exibir uma ilustração dos personagens do jogo.

Resumo visual

Você está criando um estilo visual com:

- Uma **logo do Mario** com altura definida;
- Um **botão chamativo**, estiloso, com animação ao passar o mouse;
- Uma **imagem do Mario e Luigi** em destaque.

```
body {  
    height: 100vh;  
}  
  
.caixa-mae {  
    display: flex;  
    align-items: center;  
    justify-content: space-around;  
    padding: 100px;  
    height: 100vh;  
}  
  
.caixa-principal {  
    width: 40%;  
}  
  
.caixa-video {  
    position: fixed;  
    top: 0;  
    z-index: -1;  
}
```

Esse código CSS está organizando o layout principal da sua página, provavelmente dividida entre um conteúdo textual e um vídeo de fundo.

body

```
body {  
    height: 100vh;  
}
```


- Define que o body (corpo da página) terá **100% da altura da janela do navegador**.
- 100vh significa *100% da Viewport Height* (altura da tela visível do navegador).

.caixa-mae

```
.caixa-mae {  
  display: flex;  
  align-items: center;  
  justify-content: space-around;  
  padding: 100px;  
  height: 100vh;  
}
```

Essa é uma **div contêiner principal**, com layout em **flexbox**. Vamos por partes:

Propriedade	O que faz
<code>display: flex;</code> caixa flexível	Ativa o flexbox , permitindo distribuir os filhos lado a lado
<code>align-items: center;</code> alinhar-itens: centro	Alinha os itens verticalmente ao centro
<code>justify-content: space-around;</code> justificar-conteúdo: espaço-ao-redor;	Distribui os elementos com espaço ao redor
<code>padding: 100px;</code>	Dá um espaço interno de 100px em todos os lados
<code>height: 100vh;</code>	Ocupa toda a altura da tela

 **Resultado:** os elementos dentro dessa "caixa-mãe" ficarão lado a lado, centralizados, bem distribuídos e com espaçamento interno generoso.

.caixa-principal

```
.caixa-principal {  
  width: 40%;  
}
```

- Define que a largura dessa caixa será **40% da largura da tela**.
- Provavelmente é onde está o conteúdo principal: textos, botões, imagens...

.caixa-video

```
.caixa-video {  
  position: fixed;  
  top: 0;  
  z-index: -1;  
}
```

Essa classe serve para **exibir um vídeo como fundo fixo** da página. Explicação:

Propriedade	O que faz
position: fixed	Faz o elemento ficar fixo na tela , mesmo ao rolar a página
top: 0	Cola o topo do vídeo ao topo da janela
z-index: -1	Joga o vídeo atrás de todos os outros elementos (em camadas)

 Geralmente usada para criar um **efeito de vídeo de fundo** bem estiloso.

Resumo geral:

- body: define altura total da tela.
- .caixa-mae: estrutura flexível e centralizada da página.
- .caixa-principal: área do conteúdo principal (40% de largura).
- .caixa-video: vídeo de fundo que fica fixo e atrás de tudo.

```

video {
  min-height: 100%;
  min-width: 100%;
  position: fixed;
  top: 0;
}

.mascara {
  height: 100%;
  width: 100%;
  background: linear-gradient(109deg, rgba(10, 12, 16, 0.99) 15%, rgba(10, 12, 16, 0.7) 50%, rgba(10, 12, 16, 0.99) 85%);
  position: fixed;
  top: 0;
}

p {
  position: relative;
  margin-bottom: 10px;
  color: white;
  font-size: 15px;
}

.Link-whatsapp img {
  height: 50px;
  position: fixed;
  right: 20px;
  bottom: 20px;
}

```

Esse trecho de CSS é responsável por:

1. Exibir um **vídeo de fundo**;
2. Aplicar uma **máscara escura em gradiente** por cima dele (pra deixar o conteúdo legível);
3. Estilizar textos (<p>) e um botão de **WhatsApp flutuante**.

Vamos ver parte por parte:

video

```
video {  
  min-height: 100%;  
  min-width: 100%;  
  position: fixed;  
  top: 0;  
}
```

- Garante que o vídeo:
 - **Preencha toda a tela** com `min-height` e `min-width`;
 - **Fique fixo** no topo da página com `position: fixed` e `top: 0`;
- Isso cria o famoso efeito de **vídeo de fundo**, que não se move quando a página rola.

.mascara

```
.mascara {  
  height: 100%;  
  width: 100%;  
  background: linear-gradient(109deg, rgba(10, 12, 16, 0.99) 15%,  
    rgba(10, 12, 16, 0.7) 50%, rgba(10, 12, 16, 0.99) 85%);  
  position: fixed;  
  top: 0;  
}
```

- `.mascara` é um **elemento sobreposto ao vídeo**, servindo como uma camada de "escurecimento" com **gradiente transparente**.
- Isso melhora a **visibilidade do conteúdo por cima do vídeo** (textos, botões).
- `position: fixed` e `top: 0` colocam ela **em cima de tudo**, ocupando 100% da tela.

p

```
p {  
  position: relative;
```

```
margin-bottom: 10px;
color: white;
font-size: 15px;
}
```

- Define o estilo de todos os parágrafos:
 - `position: relative` é útil se você for posicionar algo em relação ao `<p>`;
 - `margin-bottom: 10px` cria espaçamento entre os parágrafos;
 - `color: white` e `font-size: 15px` definem cor e tamanho da fonte.

`.link-whatsapp img`

```
.link-whatsapp img {
  height: 50px;
  position: fixed;
  right: 20px;
  bottom: 20px;
}
```

- Estiliza a imagem (ícone do WhatsApp) dentro de uma div/classe `.link-whatsapp`:
 - **Altura de 50px** (deixa a imagem num tamanho bom pra clicar);
 - `position: fixed` posiciona o ícone **fixo na tela** (mesmo ao rolar);
 - `right: 20px` e `bottom: 20px` colocam o ícone no **canto inferior direito** da tela.

Resumo geral

Esse CSS cria um **layout moderno e interativo** com:

- Um **vídeo de fundo** fixo;
- Uma **máscara escura com gradiente** pra legibilidade;
- Textos brancos bem posicionados;
- Um botão de **WhatsApp flutuante**, fácil de acessar.

```

.header {
  display: flex;
  align-items: center;
  gap: 30px;
  margin: 30px;
}

.header img {
  height: 50px;
}

.header a {
  color: white;
  font-size: 15px;
  cursor: pointer;
  text-decoration: none;
}

.header a:hover {
  color: rgb(12, 190, 146);
  font-size: 23px;
  transition: 0.5s ease-in-out;
}

```

Esse trecho de CSS estiliza o **cabeçalho** de um site com layout horizontal, ícone/logo, e links com animação no hover. Vamos por partes:

.header

```

.header {
  display: flex;
  align-items: center;
  gap: 30px;
  margin: 30px;
}

```

Esse bloco define o **layout da barra de navegação/cabeçalho**:

Propriedade	Explicação
<code>display: flex;</code>	Os elementos filhos (como imagem e links) ficam lado a lado

<code>align-items:</code>	Alinha os itens verticalmente ao centro da altura da
<code>center;</code>	<code>.header</code>
<code>gap: 30px;</code>	Cria um espaçamento de 30px entre os itens (logo e links)
<code>margin: 30px;</code>	Espaço externo de 30px em todos os lados do cabeçalho

`.header img`

```
.header img {
  height: 50px;
}
```

- Define que qualquer imagem dentro do `.header` terá **50px de altura**.
- Geralmente é usado para um **logo da marca**.

`.header a`

```
.header a {
  color: white;
  font-size: 15px;
  cursor: pointer;
  text-decoration: none;
}
```

Estiliza todos os **links de navegação** no cabeçalho:

Propriedade	Explicação
<code>color: white;</code>	Define a cor branca dos links
<code>font-size: 15px;</code>	Define o tamanho da fonte
<code>cursor: pointer;</code>	Mostra o cursor em forma de mãozinha (indica clique)
<code>text-decoration: none;</code>	Remove o sublinhado padrão dos links

✦ .header a:hover

```
.header a:hover {  
  color: rgb(12, 190, 146);  
  font-size: 23px;  
  transition: 0.5s ease-in-out;  
}
```

Esse trecho define o **efeito ao passar o mouse sobre os links** (hover):

Efeito	Resultado
color: rgb(12, 190, 146);	Muda a cor do link para um verde água
font-size: 23px;	Aumenta o tamanho do texto
transition: 0.5s ease-in-out;	A transição da animação ocorre de forma suave e gradual durante meio segundo

☑ Resumo

Essa parte do CSS cria uma **barra de navegação moderna** com:

- Layout em linha com flex;
- Logo à esquerda e links ao lado;
- Efeitos de hover chamativos (muda cor e tamanho dos links ao passar o mouse).

```

.formulario-fale-conosco {
  background: white;
  display: flex;
  gap: 20px;
  flex-direction: column;
  position: fixed;
  left: -300px;
  top: 30%;
  padding: 20px;
  border-radius: 5px;
  transition: left 1s ease-in-out;
}

input {
  height: 40px;
  border-radius: 5px;
  border: 1px solid gray;
  padding-left: 5px;
  outline-color: rgb(0, 165, 36);
}

textarea {
  width: 270px;
  height: 100px;
  border-radius: 5px;
  border: 1px solid gray;
  padding-left: 5px;
  outline-color: rgb(0, 165, 36);
}

```

Esse código CSS estiliza um **formulário de contato** (`.formulario-fale-conosco`) e seus campos (`input` e `textarea`). Vamos por partes:

`.formulario-fale-conosco`

```

.formulario-fale-conosco {
  background: white;
  display: flex;
  gap: 20px;
  flex-direction: column;
  position: fixed;

```




```

    left: -300px;
    top: 30%;
    padding: 20px;
    border-radius: 5px;
    transition: left 1s ease-in-out;
}

```

Esse bloco define o **estilo geral do formulário**:

Propriedade	O que faz
<code>background: white;</code>	Define o fundo branco do formulário
<code>display: flex;</code>	Usa Flexbox para organizar os elementos internos
<code>gap: 20px;</code>	Cria um espaçamento de 20px entre os campos do formulário
<code>flex-direction: column;</code>	Organiza os elementos verticalmente
<code>position: fixed;</code>	Mantém o formulário fixo na tela , mesmo ao rolar
<code>left: -300px;</code>	Inicialmente, esconde o formulário à esquerda da tela (fora da visão)
<code>top: 30%;</code>	Alinha o topo do formulário a 30% da altura da tela
<code>padding: 20px;</code>	Adiciona espaço interno entre as bordas e os elementos dentro
<code>border-radius: 5px;</code>	Deixa as bordas arredondadas
<code>transition: left 1s ease-in-out;</code>	Permite um efeito suave ao mostrar/ocultar o formulário movendo ele horizontalmente

 Com esse estilo, quando o formulário for "mostrado", o `left` pode ser alterado por JavaScript para 50%, e ele "desliza" para o centro da tela com animação.

input

```

input {
    height: 40px;
    border-radius: 5px;
    border: 1px solid gray;
    padding-left: 5px;
    outline-color: rgb(0, 165, 36);
}

```

```
}
```

Estiliza todos os **campos de entrada (input)** do formulário:

Propriedade	O que faz
height: 40px;	Define a altura dos campos
border-radius: 5px;	Borda arredondada
border: 1px solid gray;	Borda fina cinza
padding-left: 5px;	Espaço interno à esquerda, para o texto não colar na borda
outline-color: rgb(0, 165, 36);	Muda a cor da borda de foco (quando o usuário clica) para um verde vibrante

textarea

```
textarea {  
  width: 270px;  
  height: 100px;  
  border-radius: 5px;  
  border: 1px solid gray;  
  padding-left: 5px;  
  outline-color: rgb(0, 165, 36);  
}
```

Muito parecido com o input, mas com ajustes para acomodar mais texto:

- **Largura** de 270px;
- **Altura** maior (100px);
- Mesmo estilo visual (bordas arredondadas, foco verde, padding).

Resumo geral

Esse código cria um **formulário lateral oculto** que desliza para o centro quando acionado, com campos bem estilizados e responsivos. A estética é limpa e moderna, com foco em usabilidade.

```

.mascara-formulario {
  visibility: hidden;
  position: fixed;
  top: 0;
  width: 100vw;
  height: 100vh;
  background: linear-gradient(109deg, rgba(10, 12, 16, 0.99) 15%, rgba(10, 12, 16, 0.7) 50%, rgba(10, 12, 16, 0.99) 85%);
  transition: visibility 1s ease-in-out;
}

@media(max-width: 1100px){

  p {
    display: none;
  }

  .caixa-mae {
    flex-direction: column;
    justify-content: flex-start;
    padding: 10px;
  }

  .caixa-principal {
    display: flex;
    align-items: center;
    flex-direction: column;
  }

  .imagem-mario-luigi {
    width: 80vw;
    height: auto;
  }
}

```


Esse trecho de código contém duas partes principais: a **máscara do formulário** (`.mascara-formulario`) e um **media query** que adapta o layout para telas menores (como celulares ou tablets). Vamos analisar por partes:

.mascara-formulario

```
.mascara-formulario {
  visibility: hidden;
  position: fixed;
  top: 0;
  width: 100vw;
  height: 100vh;
  background: linear-gradient(109deg, rgba(10, 12, 16, 0.99) 15%,
  rgba(10, 12, 16, 0.7) 50%, rgba(10, 12, 16, 0.99) 85%);
  transition: visibility 1s ease-in-out;
}
```

Essa classe representa a **camada de fundo escura** que aparece quando o formulário de contato é exibido.

Propriedade	O que faz
visibility: hidden;	A máscara começa invisível
position: fixed;	Fixa a máscara no topo da tela, sem se mover com o scroll
top: 0;	Posiciona no topo da página
width: 100vw;	Largura total da janela (viewport)
height: 100vh;	Altura total da janela
background: linear-gradient(...)	Aplica um degradê escuro para criar um efeito de "escurecimento" do fundo
transition: visibility 1s ease-in-out;	Animação suave ao mudar a visibilidade

 **Resumo:** essa máscara serve para **escurecer o fundo da página** quando o formulário aparece, trazendo foco ao usuário. A visibilidade é controlada via JavaScript.

@media (max-width: 1100px)

Esse trecho adapta o layout **quando a tela for menor que 1100px de largura**, ou seja, para **responsividade**.

1. Esconde o parágrafo:

```
p {  
  display: none;  
}
```

- Esconde todos os <p> da página em telas menores (para simplificar o layout).

2. Ajusta .caixa-mae:

```
.caixa-mae {  
  flex-direction: column;  
  justify-content: flex-start;  
  padding: 10px;  
}
```

- **Muda a direção do Flexbox para coluna**, empilhando os elementos (em vez de lado a lado).
- Alinha os conteúdos ao topo com `justify-content: flex-start`.
- Reduz o `padding` para 10px (mais adequado para telas pequenas).

3. Ajusta .caixa-principal:

```
.caixa-principal {  
  display: flex;  
  align-items: center;  
  flex-direction: column;  
}
```

- Centraliza os itens horizontalmente.
- Empilha os elementos em coluna.

4. Ajusta imagem:

```
.imagem-mario-luigi {  
  width: 80vw;  
  height: auto;  
}
```

- A imagem dos personagens (Mario e Luigi, provavelmente) passa a ter **80% da largura da tela**.
- `height: auto;` garante que a **proporção da imagem seja mantida**.

☒ Resumo geral

Esse código:

- Cria uma **máscara escura** para destacar o formulário quando ativado.
- Torna o site **responsivo**, adaptando o conteúdo para celulares e tablets.
- Simplifica a interface em telas menores, ocultando parágrafos e reorganizando os elementos verticalmente.

```

.header {
  margin: 10px;
  display: flex;
  align-items: center;
  justify-content: center;
}

.header img {
  display: none;
}

.header a {
  font-size: 20px;
}
}

```

Esse trecho de código é uma **regra de estilo CSS** (provavelmente dentro de um `@media` query, apesar de o início dele estar faltando). Vamos analisar cada parte:

Estrutura esperada (com correção)

Esse trecho parece fazer parte de um `@media`, mas está faltando o início. O correto seria algo assim:

```

@media (max-width: 1100px) {
  .header {
    margin: 10px;
    display: flex;
    align-items: center;
    justify-content: center;
  }

  .header img {
    display: none;
  }

  .header a {
    font-size: 20px;
  }
}

```

```
}
```

Agora vamos explicar cada pedacinho:

.header

```
.header {  
  margin: 10px;  
  display: flex;  
  align-items: center;  
  justify-content: center;  
}
```

- `margin: 10px;` → Adiciona uma margem externa ao redor do `.header`.
- `display: flex;` → Usa o modelo Flexbox para organizar os itens.
- `align-items: center;` → Alinha verticalmente os itens no centro.
- `justify-content: center;` → Centraliza os itens **horizontalmente**.

.header img

```
.header img {  
  display: none;  
}
```

- Oculta qualquer imagem que esteja dentro do elemento `.header`.

Isso geralmente é feito para simplificar a interface em telas pequenas, como celulares.

.header a

```
.header a {  
  font-size: 20px;  
}
```


- Aumenta o tamanho da fonte dos links (<a>) dentro do .header para **20px**, tornando-os mais visíveis e clicáveis em telas menores.

☒ **Resumo**

Esse trecho de código CSS:

- É uma **versão adaptada do cabeçalho** (.header) para telas menores.
- **Centraliza** os itens do cabeçalho.
- **Oculto imagens** (como logos) no cabeçalho.
- **Aumenta o tamanho dos links**, deixando-os mais amigáveis para toque em dispositivos móveis.