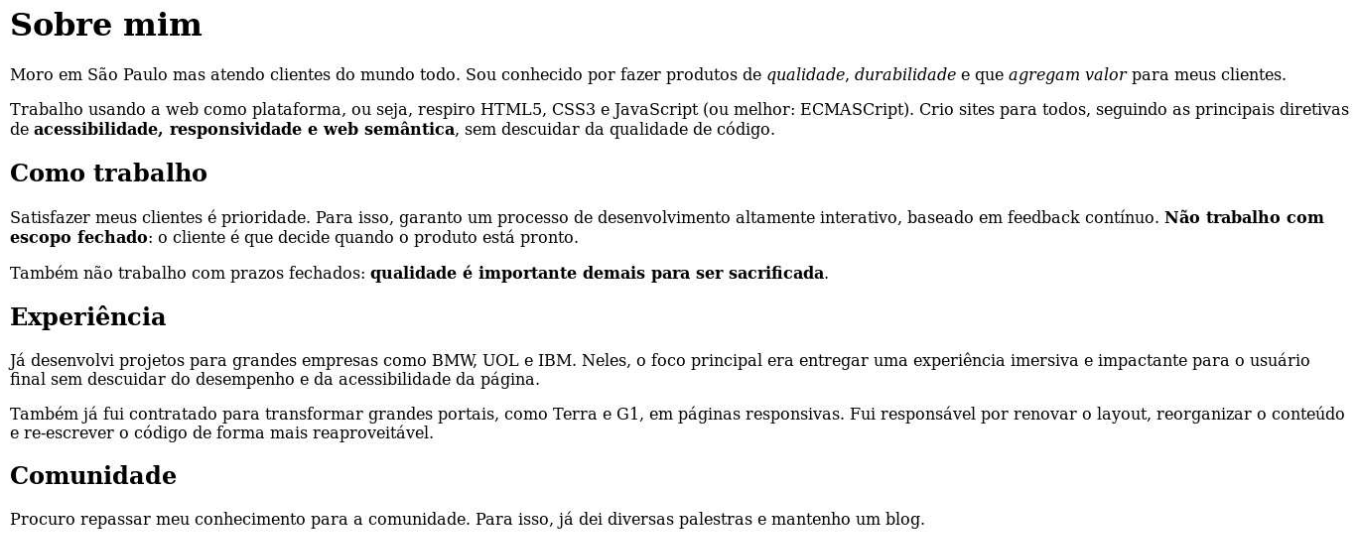
01

Construindo uma base sólida

Transcrição

Antes de começarmos a fazer os ajustes nos outros espaçamentos, repare em uma coisa: definimos os tamanhos das fontes dos títulos e subtítulos? Há espaço entre os parágrafos de texto e entre o título e as bordas da janela; fomos nós que colocamos esses espaços?

Desde a primeira aula, quando fizemos apenas o HTML, a página já tinha alguns estilos, **mesmo sem termos escrito nenhum CSS**.



Sobre mim

Moro em São Paulo mas atendo clientes do mundo todo. Sou conhecido por fazer produtos de *qualidade*, *durabilidade* e que *agregam valor* para meus clientes.

Trabalho usando a web como plataforma, ou seja, respiro HTML5, CSS3 e JavaScript (ou melhor: ECMASCript). Crio sites para todos, seguindo as principais diretivas de **acessibilidade, responsividade e web semântica**, sem descuidar da qualidade de código.

Como trabalho

Satisfazer meus clientes é prioridade. Para isso, garanto um processo de desenvolvimento altamente interativo, baseado em feedback contínuo. **Não trabalho com escopo fechado**: o cliente é que decide quando o produto está pronto.

Também não trabalho com prazos fechados: **qualidade é importante demais para ser sacrificada**.

Experiência

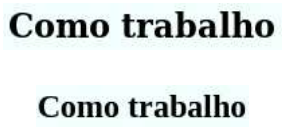
Já desenvolvi projetos para grandes empresas como BMW, UOL e IBM. Neles, o foco principal era entregar uma experiência imersiva e impactante para o usuário final sem descuidar do desempenho e da acessibilidade da página.

Também já fui contratado para transformar grandes portais, como Terra e G1, em páginas responsivas. Fui responsável por renovar o layout, reorganizar o conteúdo e re-escrever o código de forma mais reaproveitável.

Comunidade

Procuro repassar meu conhecimento para a comunidade. Para isso, já dei diversas palestras e mantenho um blog.

Ou seja, **o navegador já dá um estilo padrão** para nossas páginas. E qual o problema disso? O problema é que nem sempre queremos esse estilo padrão. E mais: como é um estilo *do navegador*, cada navegador pode fazer o seu próprio estilo, como vimos anteriormente com as fontes.



Como trabalho

Como trabalho

Nosso site precisa funcionar bem em todos os navegadores, então usar como base para o nosso CSS o estilo padrão do navegador não é uma boa ideia.

Porém, nada impede de **sobrescrevermos** o estilo padrão do navegador, como já fizemos com as fontes! No caso dos espaçamentos, vimos que podemos mexer no espaço entre eles com a propriedade `margin` e no espaço dentro deles com a propriedade `padding`. Então, se quisermos tirar o espaço que o navegador coloca entre nossos parágrafos, podemos escrever no nosso CSS:

```
p {
    margin: 0;
}
```

Podemos, também, querer tirar os tamanhos de fonte que o navegador coloca para os títulos e subtítulos:

```
h1, h2 {
    font-size: 100%; /* 100% = tamanho original da fonte do navegador */
}
```

Esses são apenas alguns exemplos. Na prática, precisamos sobrescrever *várias* regras que o navegador coloca por padrão, mesmo nos navegadores mais modernos, o que dá um bom trabalho. Para poder reaproveitar esse trabalho entre os nossos projetos, podemos até colocar essa sobrescrita num arquivo CSS separado, só para ele. Felizmente, vários desenvolvedores já tiveram essa ideia e disponibilizaram seus arquivos para que pudéssemos usar nos nossos próprios projetos.

Esse tipo de arquivo até ganhou um nome: **reset**. Atualmente, existem alguns disponíveis, com algumas diferenças entre eles. Alguns dos mais populares são:

- Eric Meyer: arquivo bem pequeno; deixa todos os elementos com mesma aparência e tamanho de fonte; usa a fonte padrão do navegador
- Normalize: arquivo um pouco maior, mas com ajustes mais finos; já define fontes e tamanhos padrão para alguns elementos; corrige diversas inconsistências entre navegadores
- YUI (Yahoo!): arquivo bem pequeno, com um efeito bem parecido com o do Eric Meyer, mas com alguns ajustes mais específicos

No nosso projeto, vamos usar o *reset* do Eric Meyer, mas você pode usar qual você achar melhor. O importante é usar um para ter **uma base sólida para desenvolver seu layout**, independente de navegador.

Como o *reset* é só mais um arquivo CSS no projeto, para usá-lo, basta baixar o arquivo e importá-lo como já fizemos com o nosso próprio CSS:

```
<link rel="stylesheet" href="reset.css">
```

Podemos, sim, ter vários arquivos CSS para uma única página HTML. Mas onde colocamos essa tag? No `<head>`? Antes ou depois do nosso CSS? Veja só: no *reset*, teremos uma regra do tipo

```
h1, h2, h3, h4, h5, h6 {
    font-size: 100%;
}
```

Já no nosso arquivo, vamos querer fazer algo do tipo

```
h1 {
    font-size: 30px;
}
```

Qual dessas duas regras o navegador deve usar? A ordem em que colocamos as tags `<link>` é importante. **O navegador lê as regras sequencialmente** e, se houver conflito entre regras, **pega a última declaração**. Ou seja, no exemplo acima, o navegador vai usar o tamanho de fonte `30px`.

Então, se queremos usar o *reset* como **base** para o nosso CSS, devemos importá-lo **antes** do nosso CSS na página:

```
<head>
    <link rel="stylesheet" href="reset.css">
    <link rel="stylesheet" href="bio.css">
</head>
```

Um pouco mais sobre conflitos no CSS

Acabamos de ver que, quando duas declarações conflitam no CSS, o navegador usa aquela que for declarada por último. Agora, o que será que acontece na seguinte situação?

```
h1 {
    font-size: 100%;
    color: red;
}

h1 {
    font-size: 30px;
    font-weight: bold;
}
```

Para entender como o navegador entende as declarações acima, precisamos entender que o conflito acontece **no elemento** onde o estilo vai ser aplicado. No exemplo acima, o estilo vai ser aplicado nos `<h1>`. Então o navegador vai fazer o seguinte:

1. Aplicar o `font-size: 100%`
2. Aplicar o `color: red`
3. Aplicar o `font-size: 30px`, sobrescrevendo o `font-size: 100%` anterior
4. Aplicar o `font-weight: bold`

Ou seja, o navegador vai simplesmente aplicando todas as regras que ele encontrar para um elemento. Se ele já mexeu numa propriedade (como o `font-size` no exemplo), não tem problema: ele mexe de novo.

A figura muda um pouco quando somos mais específicos na hora de selecionar um elemento. Imagine que agora temos o seguinte cenário: no HTML temos

```
<main>
    <h1>Título principal</h1>
</main>
```

E, no CSS, temos

```
main h1 {
    font-size: 30px;
    color: red;
}

h1 {
    font-size: 20px;
    font-weight: bold;
}
```

Quando selecionamos elementos *por hierarquia*, estamos sendo *mais específicos* do que quando selecionamos apenas pela tag. Nesse caso, o navegador começa a aplicação dos estilos partindo das regras declaradas em seletores menos específicos. Ou seja:

1. Aplica o `font-size: 20px` do seletor `h1`
2. Aplica o `font-weight: bold` do seletor `h1`
3. Aplica o `font-size: 30px` do seletor `main h1` , sobrescrevendo o `font-size: 20px`
4. Aplica o `color: red` do seletor `main h1`

Então não apenas a ordem no CSS importa, mas também a **especificidade** dos seletores que usamos. Existem várias regras para definir a especificidade de um seletor e não vale a pena entrar em detalhes aqui. O importante é saber que, quando selecionamos um elemento, ser muito ou pouco específico pode atrapalhar no desenvolvimento do projeto: podemos ser específicos demais e não conseguirmos reaproveitar código ou podemos ser específicos de menos e acabar aplicando estilos em lugares que não gostaríamos.