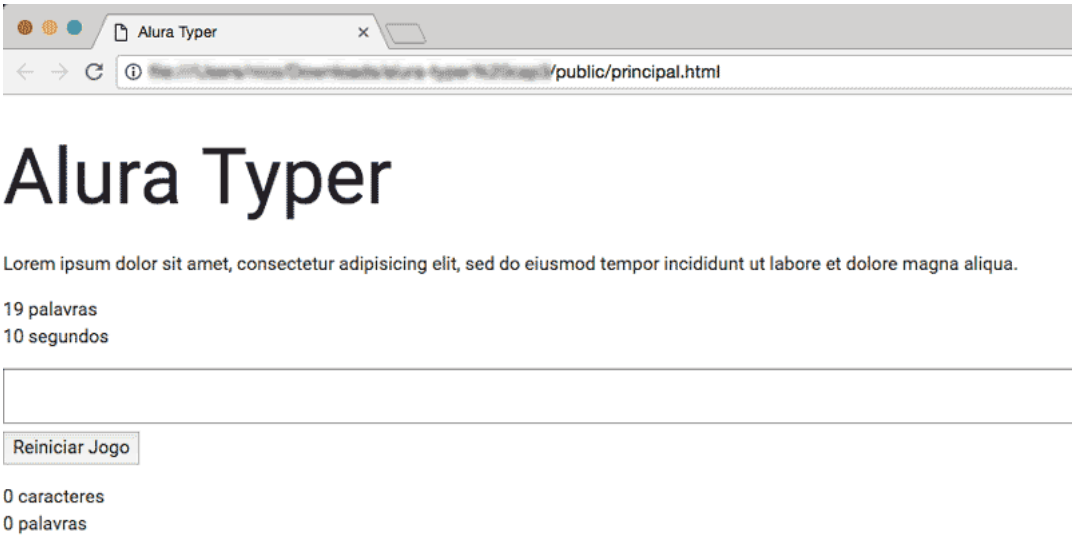
01

Estilizando o jogo

Transcrição

O visual do nosso jogo ainda está bastante simples. Se formos olhar a versão completa do jogo:



Iremos perceber alguns detalhes, como botões estilizados, campo de digitação com o fundo cinza quando o mesmo estiver desabilitado, entre outros. Mas não estamos mexendo com CSS ainda, então é justamente isso que faremos aqui neste capítulo, utilizar o CSS para estilizar o nosso jogo.

Utilizaremos um *framework*, pois o foco deste treinamento não é CSS, mas estilizaremos alguns pontos do nosso jogo com o auxílio do JavaScript, e obviamente do jQuery.

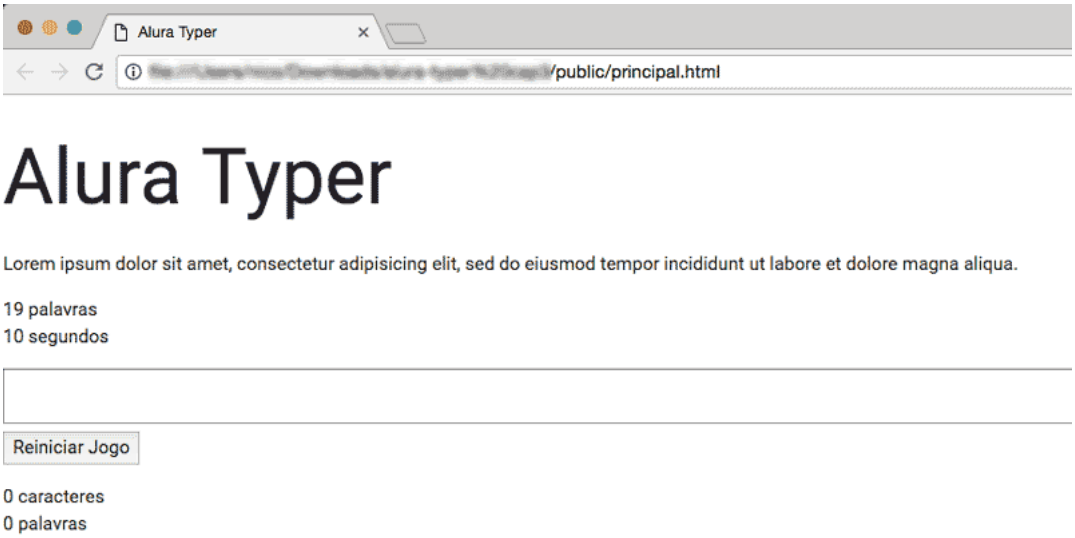
Adicionando alguns estilos ao nosso jogo

Dentro da pasta do projeto, **public/css/libs**, temos o arquivo **materialize.min.css**. O Materialize (http://materializecss.com/) é um *framework* front-end, assim como o Bootstrap, e nos auxiliará a estilizar o nosso jogo, evitando assim uma grande perda de tempo nessa tarefa.

Então vamos importá-lo na página **principal.html**, dentro da tag `<head>` :

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <title>Alura Typer</title>
    <link rel="stylesheet" href="css/libs/materialize.min.css">
</head>
```

Abra a página no navegador, e perceba já uma mudança significativa no layout da mesma:



Para estilizar ainda mais nossa página, vamos colocar todo o seu conteúdo do `body` (exceto os scripts) dentro de uma `div` com a classe `container`:

```
<body>
    <div class="container">
        <h1>Alura Typer</h1>
        <p class="frase">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmo

        <ul class="informacoes">
            <li><span id="tamanho-frase">19</span> palavras</li>
            <li><span id="tempo-digitacao">10</span> segundos</li>
        </ul>

        <textarea class="campo-digitacao" rows="8" cols="40"></textarea>
        <button id="botao-reiniciar">Reiniciar Jogo</button>

        <ul>
            <li><span id="contador-caracteres">0</span> caracteres</li>
            <li><span id="contador-palavras">0</span> palavras</li>
        </ul>
    </div>

    <script src="js/jquery.js"></script>
    <script src="js/main.js"></script>
</body>
```

Com isso, a nossa página fica alinhada no centro. Vamos centralizar também os contadores, o título e a frase, para isso adicionamos a classe `center`, mais precisamente nas duas `ul`s da nossa página, no `h1` e no `p`.

```
        <h1 class="center">Alura Typer</h1>
        <p class="frase center">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do

        <ul class="informacoes center">
            <li><span id="tamanho-frase">19</span> palavras</li>
            <li><span id="tempo-digitacao">10</span> segundos</li>
        </ul>

        <ul class="center">
```

```
            <li><span id="contador-caracteres">0</span> caracteres</li>
            <li><span id="contador-palavras">0</span> palavras</li>
        </ul>
```

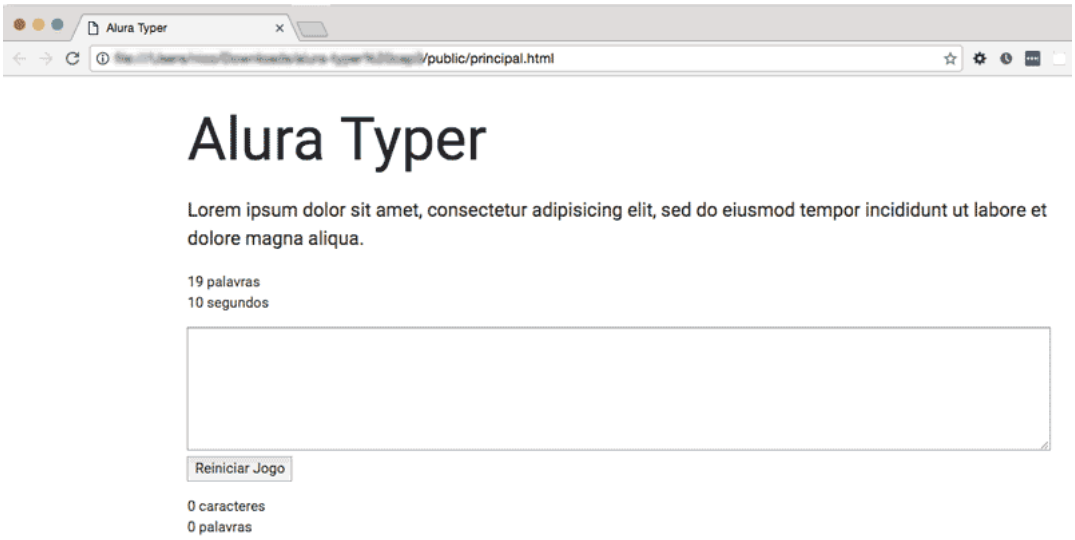
Iremos aumentar a altura do campo, alinhar a frase à esquerda, e aumentar as suas fontes. Faremos isso trabalhando dentro do arquivo `public/css/estilos.css` :

```
.campo-digitacao {
    font-size: 20px;
    height: 130px;
}

.frase {
    font-size: 20px;
}
```

Por fim, não podemos esquecer de importar esse arquivo na página `principal.html` :

```
<head>
    <meta charset="UTF-8">
    <title>Alura Typer</title>
    <link rel="stylesheet" href="css/libs/materialize.min.css">
    <link rel="stylesheet" href="css/estilos.css">
</head>
```



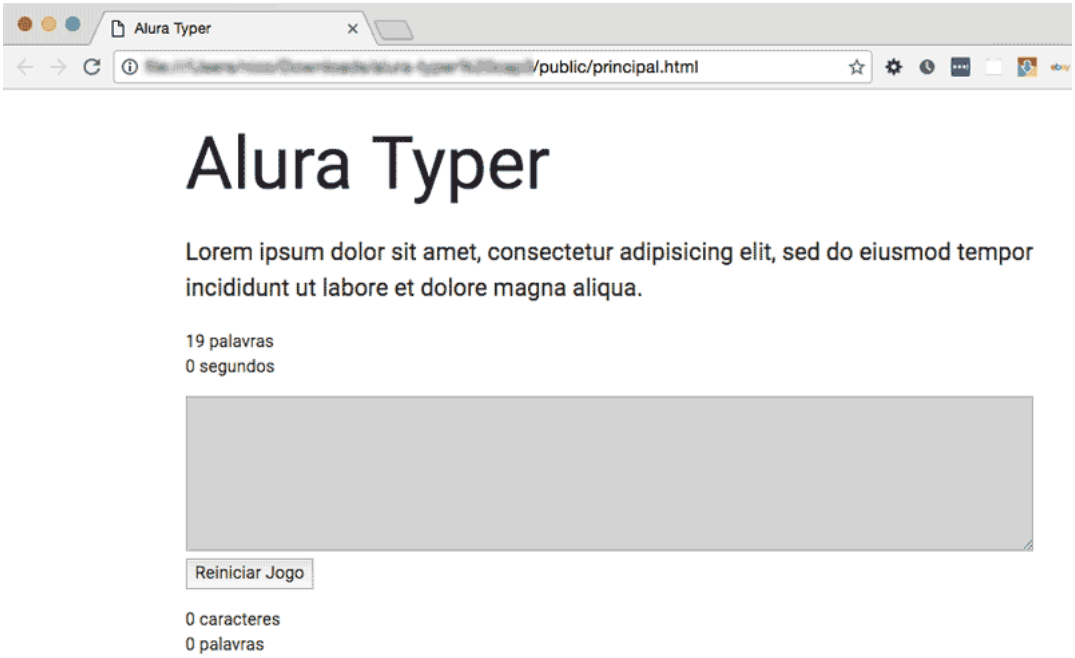
CSS com jQuery

Um outro ponto que podemos melhorar é o campo de digitação ficar com o fundo cinza quando o mesmo estiver desabilitado, quando o tempo se esgotar, para ficar bem claro para o usuário que ele não pode digitar no campo. E aí é que o jQuery nos auxilia, ele pode alterar o estilo de qualquer elemento.

No arquivo `main.js` , quando o tempo for menos que 1 (dentro da função `inicializaCronometro`), alteramos o CSS do campo utilizando a função `css()` do jQuery, passando por parâmetro a propriedade CSS queremos modificar e o seu valor, separados por vírgula:

```
function inicializaCronometro() {
    var tempoRestante = $("#tempo-digitacao").text();
    campo.one("focus", function() {
        var cronometroID = setInterval(function() {
            tempoRestante--;
            $("#tempo-digitacao").text(tempoRestante);
            if (tempoRestante < 1) {
                campo.attr("disabled", true);
                clearInterval(cronometroID);
                campo.css("background-color", "lightgray"); //novo
            }
        }, 1000);
    });
}
```

Podemos abrir a página, digitar algo e esperar o jogo finalizar:



Agora ficou bem mais claro que o usuário não pode digitar, já que o jogo foi finalizado.

Alterando o CSS no JavaScript?

Mas se o designer do jogo decidir que não quer mais que a cor do fundo fique cinza, e sim azul, teremos que procurar no `main.js` onde estamos modificando o `background-color` do elemento e alterá-lo.

Nós sabemos que não é certo mexer com o estilo de uma página, de seus elementos, no JavaScript. Se queremos estilizar, logo devemos mexer nos arquivos `.css` . Então, o que podemos fazer é, ao invés de mexer diretamente com o CSS no JavaScript, podemos **adicionar uma classe**. E no arquivo CSS nós estilizamos o campo através dessa classe.

Logo, no arquivo `estilos.css` , adicionaremos:

```
.campo-digitacao{
    font-size: 20px;
    height: 130px;
}
```

```
.frase{
    font-size:20px;
}

.campo-desativado{
    background-color: lightgray;
}
```

E no `main.js`, quando o tempo se esgotar, adicionamos essa classe `campo-desativado` no campo, através da função `addClass`:

```
function inicializaCronometro() {
    var tempoRestante = $("#tempo-digitacao").text();
    campo.one("focus", function() {
        var cronometroID = setInterval(function() {
            tempoRestante--;
            $("#tempo-digitacao").text(tempoRestante);
            if (tempoRestante < 1) {
                campo.attr("disabled", true);
                clearInterval(cronometroID);
                campo.addClass("campo-desativado");
            }
        }, 1000);
    });
}
```

Ao acessarmos a página, veremos que o comportamento continua o mesmo, porém, nosso código está mais organizado com as responsabilidades separadas.

O que aprendemos ?

- A framework Materialize.css
- Como melhorar o design da aplicação
- A função .css() do jQuery
- A função .addClass() do jQuery