09

Mãos na massa: Aplicando CSS

Começando deste ponto ?

Começando deste ponto? Você pode fazer o DOWNLOAD (https://s3.amazonaws.com/caelum-online-public/jquery-alura-typer/stages/jquery-cap-05.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

Melhorando o visual

O visual do nosso jogo ainda está bastante simples razão suficiente para aplicar um CSS, não? Como você viu no video usaremos o framework Materialize para estilizar a página que auxilia muito nessa tarefa.

Para tal:

1) Abra o arquivo `principal.html` e *adicione* dentro do `<head>` logo abaixo da tag `<title>` :

```
<link rel="stylesheet" href="css/libs/materialize.min.css">
```

Recarregue a página dentro do navegador e repare a diferença. Para sua comparação o `<head>` inteiro ficou como:

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

Obs: *O arquivo `materialize.min.css` já está dentro da pasta `public/css/libs` do seu projeto.*

2) Agora, na mesma página `principal.html` , coloque todo o conteúdo do `body` (exceto os scripts) dentro de uma `div` com a classe `container` . Adicione apenas o `div` :

```
<body>
    <div class="container">
        <!-- conteúdo da página aqui -->
    </div>

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

Novamente, recarregue a página no navegador para ver a diferença

3) Após isso, vamos aumentar a altura do campo, alinhar a frase à esquerda, e aumentar as suas fontes. Abre o arquivo `public/css/estilos.css` e coloque esse código abaixo:

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

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

4) No arquivo `principal.html` importe o `estilo.css` dentro da tag `<head>` :

```
<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"> <!-- novo -->
</head>
```

5) Para deixar mais claro que o jogo terminou, deixe o campo de digitação cinza quando o tempo esgotar. Dentro do arquivo `public/css/estilo.css` adiciona uma nova clase:

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

6) No arquivo `main.js` , quando o tempo se esgotar, devemos adicionar essa classe **`campo-desativado`** ao campo usando jQuery. Quando o usuário clicar no botão reiniciar, o campo deveria ser reativado. Para não espalhar as funções `addClass` e `removeClass` do jQuery, vamos aproveitar a função `toggleClass` que ativa e desativa respectivamente.

Abra o arquivo `main.js` e procure a função `inicializaCronometro` :

```
//dentro do arquivo main.js, dentro da função inicializaCronometro

//dentro desse if adicione apenas a linha com toggleClass
if (tempoRestante < 1) {
    campo.attr("disabled", true);
    clearInterval(cronometroID);
    campo.toggleClass("campo-desativado"); //novo
}
```

Adicione a mesma funcionalidade no final da função `reiniciaJogo` :

```
function reiniciaJogo() {
    //código omitido e não alterado
    campo.toggleClass("campo-desativado"); //novo
}
```

7) Salve os arquivo e teste no navegador. Após o tempo esgotar, o campo de digitação deve ficar cinza. Ao reiniciar volta a ser ativado.