

Adicionando ícones

Transcrição

O que está faltando em nosso jogo para realmente fique parecido com o **Alura Typer** são os ícones e botões coloridos. Você pode ver como utilizar esses ícones [aqui](http://materializecss.com/icons.html) (<http://materializecss.com/icons.html>), no site no Materialize.

O primeiro passo é importar a fonte de ícones. Nós podemos copiar o link que está no site, ou utilizar a fonte que já está no projeto, dentro da pasta `public/css/libs`. Vamos, então, importá-la dentro da tag `<head>`, na página `principal.html`:

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

Para utilizar os ícones, temos que adicionar uma tag `<i>` com as classes `small` (referente ao tamanho do ícone, há também os tamanhos `tiny`, `medium` e `large`) e `material-icons`. Dentro da tag nós colocamos o nome do ícone que queremos (a lista se encontra no site). Os ícones que utilizaremos para o contador do tamanho da frase e para o cronômetro são o `description` e `query_builder`, respectivamente.

Além disso, vamos deixar o nosso `ul` centralizado. Logo, nossa `ul` de informações ficará da seguinte forma:

```
<body>
  <ul class="informacoes center">
    <li>
      <i class="small material-icons">description</i>
      <span id="tamanho-frase">5</span> palavras
    </li>
    <li>
      <i class="small material-icons">query_builder</i>
      <span id="tempo-digitacao">10</span> segundos
    </li>
  </ul>
</body>
```

Deixaremos os ícones alinhados com o seu respectivo texto. Para isso, adicionaremos a classe `icones` às duas tags `<i>`

```
<div class="container">

  <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>
  <i class="small material-icons icones">description</i>
  <span id="tamanho-frase">5</span> palavras
</li>
<li>
  <i class="small material-icons icones">query_builder</i>
  <span id="tempo-digitacao">10</span> segundos
</li>
</ul>

```

Adicionaremos, ainda, o seguinte código no arquivo `estilos.css`:

```

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

.frase{
  font-size: 20px;
}

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

.icones{
  vertical-align: middle;
}

```

Com isso, o ícone estará perfeitamente centralizado.

Alura Typer

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

 19 palavras
 15 segundos



0 caracteres
0 palavras

Botão de reinício

O próximo passo é modificar o layout do botão de reinício de jogo. Para isso, o **Materialize** também nos auxilia: na página [Buttons](http://materializecss.com/buttons.html) (<http://materializecss.com/buttons.html>) há uma série de opções para utilizarmos. Escolheremos o **Floating**. Feito isso, iremos substituir o código do botão atual pelo código disponibilizado no site:

```
<a class="btn-floating btn-large waves-effect waves-light red">
  <i class="material-icons">add</i>
</a>
```

Para terminar, iremos inserir o id do nosso antigo botão (`botao-reiniciar`) na tag `<a>` e utilizar um ícone mais expressivo, o `restore` :

```
<a id="botao-reiniciar" class="btn-floating btn-large waves-effect waves-light red">
  <i class="material-icons">restore</i>
</a>
```

Como iremos ter mais botões futuramente, colocaremos o nosso botão atual dentro de uma `div` com a classe `botoes` :

```
<textarea class="campo-digitacao" rows="8" cols="40"></textarea>

<div class="botoes">
  <a id="botao-reiniciar" class="btn-floating btn-large waves-effect waves-light red">
    <i class="material-icons">restore</i>
  </a>
</div>
```

Podemos agora testar o nosso botão de reinício e ver que tudo continua funcionando corretamente.

Adicionando e removendo classes facilmente de um elemento

Quando clicamos no botão, conseguimos digitar novamente no campo, mas o fundo do mesmo não fica branco, podendo deixar o usuário em dúvida se o jogo realmente foi reiniciado. Vamos deixar isso bem claro, deixando o fundo do campo em branco assim que o jogo for reiniciado.

Para deixar o fundo do campo cinza, nós adicionamos a classe `campo-desativado` no campo, assim que o tempo se esgotar. E no CSS nós pintamos o fundo do campo que contiver esta classe. Então para remover o fundo, basta nós removermos a classe do campo quando o jogo for reiniciado.

Assim como o jQuery possui a função `addClass`, ele também possui a `removeClass`, logo, no `main.js`, adicionaremos `inicializaCronometro()` e `campo.removeClass()` :

```
function reiniciaJogo(){
  $("#botao-reiniciar").click(reiniciaJogo);
  function reiniciaJogo(){
    campo.attr("disabled", false);
    campo.val("");
    $("#contador-palavras").text("0");
    $("#contador-caracteres").text("0");
    $("#tempo-digitacao").text(tempoInicial);
    inicializaCronometro();
  }
}
```

```

campo.removeClass("campo-desativado");
}

```

Essa ação de adicionar e remover classes se tornou uma tarefa tão comum, que o jQuery criou uma função específica para isso, a `toggleClass()`. Ela funciona da seguinte maneira: se no momento que a função for chamada, o elemento possuir a classe, ela será removida. Mas se o elemento não possuir a classe, ela será adicionada.

Utilizaremos esta função no lugar de ambas as funções, `addClass` e `removeClass`:

```

function reiniciaJogo() {
    campo.attr("disabled", false);
    campo.val("");
    $("#contador-palavras").text(0);
    $("#contador-caracteres").text(0);
    $("#tempo-digitacao").text(tempoInicial);
    inicializaCronometro();
    campo.toggleClass("campo-desativado");
}

```

E dentro da função `inicializaCronometro()`:

```

if (tempoRestante < 1) {
    campo.attr("disabled", true);
    clearInterval(cronometroID);
    campo.toggleClass("campo-desativado");
}

```

O que aprendemos ?

- Adicionando ícones
- A boa prática de não alterar estilos diretamente no Javascript
- A função `.removeClass()` do jQuery
- A função `.toggleClass()` do jQuery