

Apresentação - PARTE 1

Transcrição

Começando deste ponto? Você pode fazer o [DOWNLOAD \(https://s3.amazonaws.com/caelum-online-public/carreira-js/stages/04-gallows.zip\)](https://s3.amazonaws.com/caelum-online-public/carreira-js/stages/04-gallows.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo.

Padrão a seguir

Chegou a hora de implementarmos a lógica de apresentação do jogo, até porque, precisamos exibir para o usuário o estado do jogo para que ele possa interagir com ele.

Usaremos a mesma estratégia que já utilizamos anteriormente. Vamos criar o arquivo `gallows/js/controller.js`.

Separando dependências, mais uma vez

Nele, criaremos a função `criaController` que receberá como parâmetro um objeto que representa nosso jogo. Lembre-se que já criamos a função `criaJogo()` que gera esse objeto para nós.

```
var criaController = function (jogo) {  
  
    var $entrada = $('#entrada');  
    var $lacunas = $('.lacunas');  
  
};
```

Nossa função declara em seu escopo as variáveis `$entrada` e `$lacunas`. O prefixo `$` é utilizado para deixar claro para quem esta lendo o código que elas armazenam objetos criados através do jQuery. Não é a toa que utilizamos `$('#entrada')` e `$('.lacunas')` para selecionarmos o campo de entrada do jogador e a lista de lacunas. Fica evidente que o primeiro é para podermos capturar o que o jogador digitou, já o segundo, não tão evidente assim, é a `<ul class="lacunas">`, aquela que receberá dinamicamente elementos do tipo `` que exibirão cada item do array de `jogo.getLacunas()`.

Da mesma maneira que fizemos no capítulo anterior, seguem as funções que precisaremos implementar. Elas lidarão com a captura da palavra secreta, exibição das lacunas com base nessa palavra e mudará o texto da propriedade `placeholder` de `$entrada` para que estejamos visualmente preparados para entrarmos na etapa de chute:

```
var criaController = function (jogo) {  
  
    var $entrada = $('#entrada');  
    var $lacunas = $('.lacunas');  
  
    // consulta jogo.getLacunas() e exibe para o usuário cada lacuna  
  
    var exibelacunas = function () {
```

```
    console.log('falta implementar');
  };

  // muda o texto do placeholder do campo de entrada
  var mudaPlaceholder = function (texto) {

    console.log('falta implementar');
  };

  // passa para jogo.setPalavraSecreta() o valor digitado pelo jogador e chama o a função `ex:

  var guardaPalavraSecreta = function () {

    console.log('falta implementar');
  };

  // faz a associação do evento keypress para capturar a entrada do usuário toda vez que ele i
  var inicia = function () {

    console.log('falta implementar');
  };

  // retorna um objeto com a propriedade inicia, que deve ser chamada assim que o controller .
  return { inicia: inicia };
};
```

O ponto de partida é a função `inicia()`. Ela deve ser capaz de adicionar o evento `keypress` em `$entrada`. Todavia, só podemos computar o evento se o usuário pressionar a tela ENTER. Além disso, ele deve ser inteligente para saber em qual etapa do jogo estamos para lidar diferentemente com a entrada do usuário, pois pode ser que ele esteja na primeira etapa (`palavraSecreta`) ou segunda etapa (`chute`).

O papel da função `inicia()`

Nesse sentido, já temos o código de `inicia()` pronto, como ponto de partida. O código inicial deve estar assim:

```
var criaController = function (jogo) {

  var $entrada = $('#entrada');
  var $lacunas = $('.lacunas');

  var exibelacunas = function () {
    console.log('falta implementar');
  };

  var mudaPlaceholder = function (texto) {

    console.log('falta implementar');
  };

  var guardaPalavraSecreta = function () {

    console.log('falta implementar');
  };
};
```

```
var inicia = function () {

    $entrada.keypress(function (event) {
        if (event.which == 13) {
            switch (jogo.getEtapa()) {
                case 1:
                    alert('etapa 1 - falta implementar');
                    break;
                case 2:
                    alert('etapa 2 - falta implementar');
                    break;
            }
        }
    });
}

return { inicia: inicia };
};
```

O arquivo `gallows/index.html` :

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="css/sprite.css">
    <link rel="stylesheet" href="css/lacuna.css">
    <title>Jogo da Forca</title>
</head>
<body>
    <div class="jumbotron">
        <h2 class="text-center">Jogo da forca</h2>
    </div>
    <div class="container">
        <div class="sprite"></div>
        <ul class="lacunas text-center"></ul>

        <div class="text-center">
            <input id="entrada"
                class="entrada"
                placeholder="Palavra secreta" autofocus>
        </div>
    </div>

    <script src="js/jquery.min.js"></script>
    <script src="js/sprite.js"></script>
    <script src="js/jogo.js"></script>
    <script src="js/controller.js"></script>
    <script>
        var sprite = createSprite('.sprite');
        var jogo = criaJogo(sprite);
        criaController(jogo);
    </script>
</body>
</html>
```

O mesmo código pode ser escrito da seguinte maneira, sem a necessidade de se declarar as variáveis `sprite` e `jogo` :

```
<script>
  criaController(criaJogo(createSprite('.sprite'))).inicia();
</script>
```

Talvez essa estética seja melhor para alguns:

```
<script>
  criaController(
    criaJogo(
      createSprite('.sprite')
    )
  ).inicia();
</script>
```

Já somos capazes de ver o alert sendo exibido toda vez que clicarmos enter.