Introdução às Tecnologias *Web*

Guião da Aula Teórico-Prática sobre JavaScript (Parte 1)

Antes da aula

Realizar as seguintes lições do tutorial Learn JavaScript do Codecademy: Introduction, Variables, Conditionals, e Functions.

Durante a aula

Comece por descarregar o ficheiro de material de apoio à aula teórico-prática, o qual contém dois ficheiros, `index.html` e `comportamento.js`, estando este último na pasta `scripts`. No cabeçalho do documento HTML vem uma referência para o ficheiro de JavaScript que faz com que o *browser* carregue e automaticamente execute o código aí existente.

Exercício 1: Abra o ficheiro `index.html` num editor de código e compare o conteúdo com o que é mostrado se o carregar num *browser*. Reparou que há texto a menos no ficheiro HTML?

Exercício 2: Inspecione o ficheiro `comportamento.js` e descubra se o texto a mais que aparece no *browser* está, afinal, no código JavaScript. Leia os comentários no código para perceber o comportamento do *browser* quando o `index.html` é carregado, em especial a explicação sobre a primeira instrução do *script* que é executada e quando é invocada a função `principal()`.

Neste ponto da aula, deverá ficar claro que, mais do que simplesmente mostrar o conteúdo de um ficheiro HTML, um *browser* pode executar código que altera esse conteúdo. É importante perceber que o ficheiro HTML permanece inalterado, sendo todo o processamento realizado sobre uma árvore DOM *(Document Object Model)*, que é uma representação em memória da informação existente no documento HTML, acessível através da variável global `document`.

Na função `principal()` de `comportamento.js` vem um exemplo de escrita de uma mensagem na consola do *browser*, sendo esta uma ferramenta essencial para testar código JavaScript. Assim, os próximos exercícios vão requerer o uso da consola do *browser*, não apenas para ver mensagens, mas também para introduzir comandos.

Exercício 3: Edite o ficheiro `comportamento.js` e codifique a função `maiorDeDoisNumeros()`, com dois parâmetros, que aceita dois números e devolve o maior deles. Carregue novamente o `index.html` no *browser,* para que este tenha em conta o código da nova função. Verifique que é produzido o resultado esperado introduzindo comandos na consola, tais como: `maiorDeDoisNumeros(1, 2)` e `maiorDeDoisNumeros(20, 10)`.

Para o exercício que vem a seguir, é útil saber que um *browser* pode ser programado para pedir dados ao utilizador, e uma forma de o fazer é usando a função `Window.prompt()`. Para invocar esta função basta escrever `prompt()`, sendo implicitamente assumida a variável global `window`, pertencente à interface `Window`. A função recebe a mensagem do pedido de dados e devolve o texto que o utilizador tiver digitado.

Exercício 4: No ficheiro `comportamento.js`, codifique a função `processaDoisNumeros()` que pede dois números ao utilizador, calcula o maior deles, tirando partido da função `maiorDeDoisNumeros()` criada no exercício 3, e devolve texto na seguinte forma: `os números digitados foram X e Y, e o maior é Z`. Para testar a função, primeiro recarregue o `index.html`, e depois invoque-a na consola do *browser*, fornecendo, por exemplo, os valores 9 e 100.

Caso tenha obtido resultados errados nos testes de `processaDoisNumeros()`, pode estar a suceder que os dados introduzidos pelo utilizador não estejam a ser interpretados como números mas sim como texto. Por exemplo, a condição `"9" > "100"` é verdadeira, pois a comparação é feita uma posição de cada vez, e logo na primeira vem `"9" > "1"`. Para garantir que os números são bem interpretados, pode aplicar a função `Number()` ao resultado de `prompt()`.

Exercício 5: Atualize a função `processaDoisNumeros()` para que no texto resultante apareça também o menor dos dois números, passando a ser na forma: `os números digitados foram X e Y, o maior é Z, e o menor é W`. Para obter o menor de dois números, pode criar uma nova função ou então invocar `maiorDeDoisNumeros()` com trocas simples de sinal nos argumentos que são fornecidos e no resultado que é devolvido. Em qualquer dos casos, certifique-se que a documentação é apropriada e faça testes usando a consola do *browser*.

Com os exercícios 3 a 5 da aula, ganhou experiência essencial a programar em JavaScript e a utilizar a consola para invocar funções e obter os seus resultados. Contudo, quando navegamos num *site*, toda a interação com o utilizador é feita através da janela que mostra as páginas HTML, não sendo necessário usar a consola. Assim, o próximo e último exercício recupera o propósito dos exercícios 1 e 2, de perceber as formas de programar um *browser* para mostrar conteúdos e tratar dados introduzidos pelo utilizador.

Exercício 6: Acrescente código à função `principal()` para que, quando o *browser* termina o carregamento de `index.html`, seja também invocada a função `processaDoisNumeros()` e a mensagem de resultado desse processamento apareça no texto do parágrafo com identificador `resultado` da página HTML.

Para concluir este guião, fica um apontamento sobre a notação usada nos comentários que estão logo antes das funções `principal()` e `maiorDeDoisNumeros()` do material de apoio à aula. Trata-se da notação JSDoc, cujos comentários começam sempre com `/**` (são dois asteriscos seguidos) e que ajudam a produzir automaticamente documentação, como, por exemplo, uma página HTML com as descrições, parâmetros (ver `@param`), e resultados (`@returns`) de todas as funções existentes num *script*. Alguns editores, como o Visual Studio Code, reconhecem a notação JSDoc e, enquanto escrevemos o código de invocação de uma função, mostram uma *tooltip* com a respetiva documentação (tente com a `maiorDeDoisNumeros()`), facilitando a escrita e manutenção de código e aumentando a produtividade.