Interface Pessoa-Máquina

Licenciatura em Engenharia Informática

Ficha Prática #05

Rafael Braga
d13414@di.uminho.pt

Daniel Murta
d6203@di.uminho.pt

José Creissac Campos
jose.campos@di.uminho.pt

(v. 2024)

Conteúdo

1 Objetivos

1. Praticar a utilização de Media Queries e Javascript.

2 CSS Media Queries e Javascript

Esta ficha prática apresenta exercícios sobre *media queries* e JavaScript, duas técnicas importantes para o desenvolvimento de aplicações web.

Media queries são uma funcionalidade do CSS3 que permite testar e consultar valores do *viewport* e características do navegador ou do dispositivo, para aplicar estilos CSS condicionalmente. As *media queries* são usadas na regra `@media` do CSS e permitem adaptar o *layout* e conteúdo de uma página web às diferentes características dos ecrãs e dispositivos, criando um design responsivo. Por exemplo, podemos mudar a organização do conteúdo na página em função da largura do ecrã.

Javascript é uma linguagem de programação interpretada com um sistema de tipos dinâmico e fraco (não verifica nem impõe a compatibilidade entre os tipos de dados das variáveis, constantes, expressões ou funções, e o tipo das variáveis pode ser alterado em tempo de execução). A linguagem permite adicionar interatividade, dinamismo e lógica a uma aplicação web. Javascript pode ser usado para manipular o DOM (*Document Object Model*), enviar e receber dados, criar animações, validar formulários, etc. Nesta ficha irá praticar a manipulação do DOM.

3 Exercícios

Resolva os seguintes exercícios.

3.1 Responsive Blog

Considere o website referente a uma rede de bloggers que foi apresentado na Ficha 4. Uma implementação desse website é fornecida com esta ficha. Um dos problemas deste website é que não se ajusta a diferentes tamanhos de ecrã (ou seja, não é *responsive*). O objetivo deste exercício consiste em tornar este website ajustável a diferentes dimensões de ecrã através do uso de *CSS Media Queries*. Para tal, implemente as seguintes etapas:

1. Ajuste o *layout* base do website para uma grelha com apenas 1 coluna. Os elementos referentes ao conjunto de histórias, a galeria de top bloggers e a

lista de categorias deverão passar a ter uma margem de 20 pixeis à esquerda e à direita. Aplique estas regras a dimensões iguais ou inferiores a 850 pixeis.

2. Considere o elemento com o id "menu" presente na *navigation bar*. Ao analisar as regras CSS para este elemento, pode-se verificar que este elemento não é renderizado no website através da propriedade `display: none`. Acrescente um novo conjunto de regras CSS para ecrãs com dimensões iguais ou inferiores a 500 pixeis que deverão produzir os seguintes efeitos (o resultado deverá ser igual ao da Figura 1):

   - Deverá deixar de renderizar todos os elementos da navigation bar com exceção do logótipo.
   - Deverá renderizar o elemento "menu".
   - O layout base deverá ter uma largura mínima de 300 pixeis.

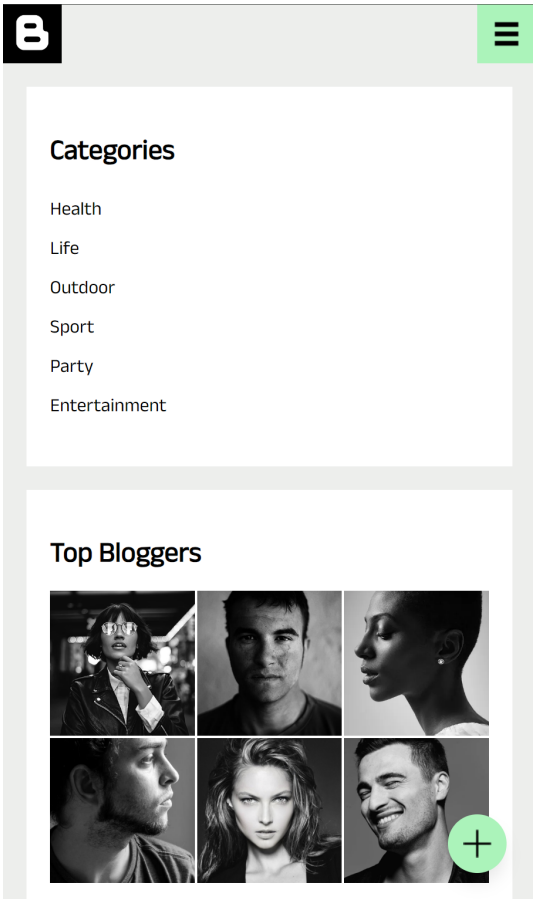

Figura 1: Website de bloggers – versão *responsive*

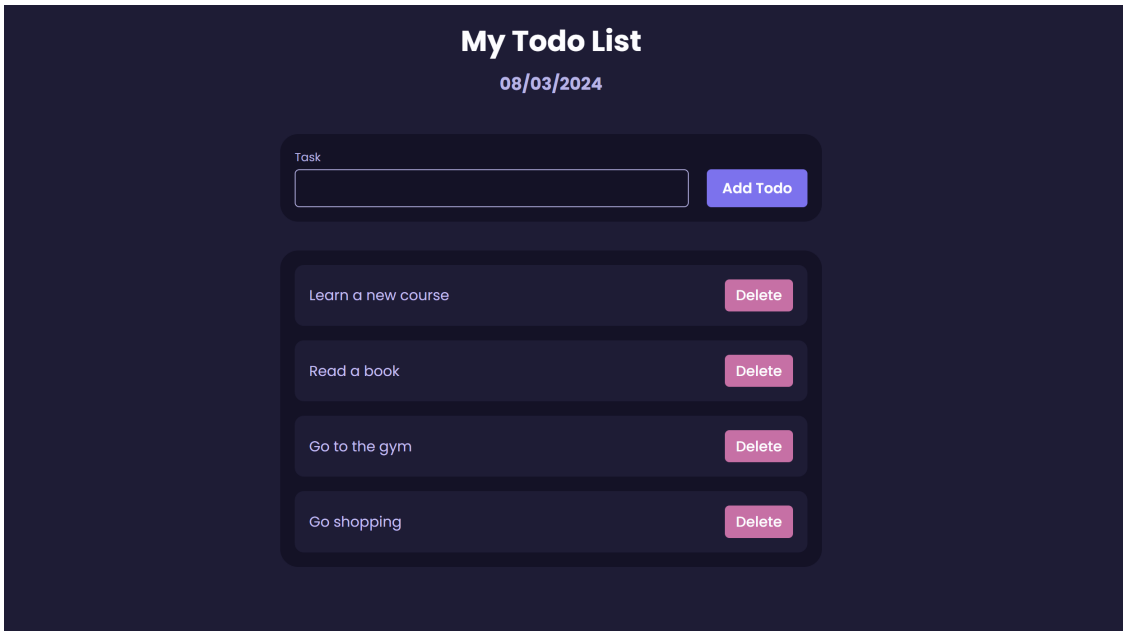

Figura 2: Website de lista de todos

3.2 Lista de *Todos*

Considere o website apresentado na Figura 2, que representa uma lista de tarefas (*todos*) a serem realizadas num dia. Considere também a implementação base fornecida nesta ficha que contém os seguintes elementos:

- O ficheiro "index.html" que contém toda a estrutura do website.
- O ficheiro "styles.css" que contém o conjunto de regras CSS que permite chegar ao aspeto do website apresentado.
- O ficheiro "todo.js" que contém um array inicial de *todos* e onde se deverá acrescentar todo o comportamento do website.

O objetivo deste exercício consiste em dar comportamento ao website de modo a que um utilizador consiga acrescentar uma lista de tarefas (únicas). Deverá também ser possível ir removendo tarefas à medida que um utilizador as termine. Para tal implemente as seguintes etapas:

1. Adicione a data atual ao elemento `<h3>` com o id "list-date". Deverá fazer isto quando todo o conteúdo do DOM estiver carregado.
2. Adicione um *event listener* para o evento `onsubmit` à `<form>` com o id "todo-form". Tenha em atenção que este evento de submição faz, por omissão, o *refresh* de toda a página. Como fazer para prevenir este comportamento?

O *event listener* deverá acionar uma função que adicione a tarefa escrita pelo utilizador ao array de *todos*. No entanto, a tarefa só deverá ser adicionada se contiver texto (não deve consistir apenas em espaços) e se não existir já no array de *todos*. Caso já exista no array de *todos*, deverá ser mostrada uma mensagem de erro ao utilizador (utilize a função *alert*). Verifique o funcionamento da função imprimindo o array na consola (através da função `console.log`). Pode ver a consola através das *developer tools* do seu *browser*. No final, o *event listener* deverá limpar o texto introduzido pelo utilizador.

3. Crie uma função que renderiza o array de *todos*, tal como ilustrado na Figura 2. A função deverá começar por remover todos os elementos já existentes no elemento com o id "todo-list". De seguida, para cada tarefa no array de *todos* deverá criar um elemento `<li>` que deverá conter um elemento `<p>` onde será mostrada a tarefa, e um `<button>` que deverá conter a label "Delete". Além disso, o elemento `<li>` adicionado deverá ter a classe "todo-list-item". Cada elemento `<li>` deverá ser adicionado ao elemento com o id "todo-list".

4. Adicione um *click listener* a cada `<button>` criado na etapa anterior. Este evento deverá acionar uma função que remove do array de todos a tarefa correspondente ao elemento clicado. No final a interface deverá refletir estas mudanças.

3.3 Jogo do Galo

Crie uma página web para jogar ao jogo do Galo.

1. Crie a grelha para jogar, como na Figura 3;
   (a) A grelha deverá ser um quadrado de 600px de lado;
   (b) Em cada célula da grelha, o texto deverá ser centrado vertical e horizontalmente.
   (c) Ao passar o rato por cima de uma célula, a mesma deverá mudar de cor de fundo.
   (d) As linhas (a preto) delimitadoras da grelha deverão ser desenhadas;
2. Centre a grelha no meio do ecrã (horizontal e verticalmente).
3. Implemente o evento de click em cada uma das células da grelha:
   (a) Se for a vez das cruzes/bolas, ao clicar na célula a mesma deverá ficar preenchida com um X/O maiúsculo;
   (b) Assuma que sempre que o jogo começa, jogam primeiro as cruzes;

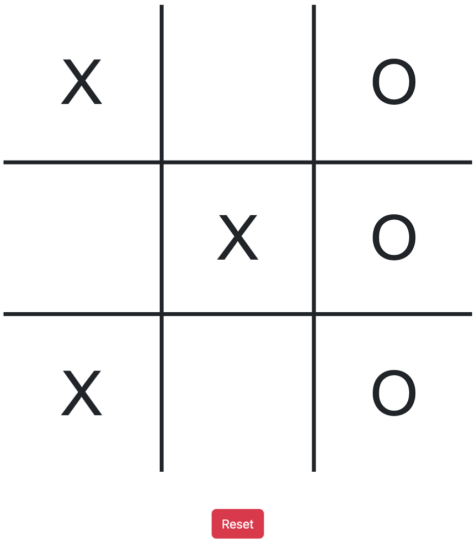

Figura 3: Jogo do Galo

4. Garanta que, se o utilizador tentar clicar numa célula já preenchida, é devolvida uma mensagem de erro "Célula já preenchida". (Dica: use a função *window.alert*)

5. Adicione e implemente o botão de reset para começar o jogo de novo.

6. Implemente uma função que determine se já há um vencedor para o jogo. De cada vez que seja feita uma jogada:

   (a) Se o jogo já tiver terminado antes, deverá ser devolvida a seguinte mensagem "Jogo Terminado! Faça reset para recomeçar."

   (b) Se o jogo terminar após a jogada, deverá ser devolvida uma mensagem indicando o vencedor, caso o haja, ou "Empate"em caso contrário.