Interface Pessoa-Máquina

Licenciatura em Engenharia Informática

Ficha Prática #04

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 HTML e CSS.

2 HTML e CSS

HTML e CSS são duas linguagens fundamentais para o desenvolvimento de páginas web.

2.1 HTML

HTML (*HyperText Markup Language*) é a linguagem padrão utilizada para criar e estruturar conteúdos na Web. Utiliza um sistema de elementos e tags para definir os componentes de uma página Web, tais como títulos, parágrafos, ligações e imagens. O HTML funciona como a espinha dorsal do desenvolvimento Web, fornecendo a estrutura básica que pode ser melhorada com CSS para estilo e JavaScript para interatividade.

Um conceito fundamental do HTML é o *semantic markup*, que enfatiza a utilização de tags HTML para transmitir o significado do conteúdo e não apenas a estruturação do mesmo. Esta abordagem é relevante para garantir a acessibilidade dos conteúdos e otimizar o desempenho dos motores de busca. Estruturar o conteúdo HTML de forma a realçar o seu valor semântico (o que os elementos significam) é também relevante para adicionar posteriormente o estilo de apresentação e o comportamento. Uma abordagem típica é ter elementos para:

- Nome da aplicação (tag `<header>`)
- Barra de navegação (tag `<nav>`)
- Conteúdo principal (tag `<main>`)
- Conteúdo lateral/adicional (tag `<aside>`)
- Rodapé (tag `<footer>`)

2.2 CSS

CSS (*Cascading Style Sheets*) é a linguagem de estilos que define a aparência e o *layout* de um documento web, através de um conjunto de regras de estilo. Estas regras de estilo são compostas por seletores, que indicam quais elementos HTML serão

estilizados, e por declarações, que indicam que propriedades serão aplicadas. As propriedades permitem definir aspetos como cores, fontes, margens, etc.

Um função importante do CSS é definir o posicionamento dos elementos na página (o *layout*). Particularmente útil é o *layout* Flexbox, que permite organizar os elementos numa caixa flexível, que se adapta ao espaço disponível. Para usar Flexbox, é preciso definir a propriedade CSS `display` como `flex` ou `inline-flex`, no elemento que atua como *flex container*. Os filhos diretos do *flex container* (os *flex items*) irão ser colocados sequencialmente na direção definida no *flex container* (horizontal ou vertical) e podem ter as suas próprias propriedades *flex* para controlar o seu comportamento. No entanto, a utilização de Flexbox pode tornar-se complexa para *layouts* bidimensionais, como os *layouts* de páginas inteiras.

CSS Grid é um módulo de *layout* que permite criar *layouts* bidimensionais complexos e responsivos com facilidade. Fornece um sistema de *layout* baseado numa grelha, permitindo um controlo preciso sobre as linhas e colunas dessa grelha. O conteúdo da página é colocado nas células da grelha, de modo a ser posicionado e dimensionado de forma flexível.

Configurar um CSS Grid envolve alguns passos fundamentais. Primeiro, é preciso selecionar um elemento para funcionar como *grid container*. Isso é feito definindo a propriedade CSS `display` como `grid` nesse elemento. Em seguida, utilizam-se as propriedades `grid-template-columns` e `grid-template-rows` para especificar o número e o tamanho das colunas e linhas da grade. Para posicionar itens dentro da grade, podem usar-se propriedades como `grid-column` e `grid-row`, que permitem controlar a localização exata de cada item. No entanto, para uma organização mais intuitiva, a propriedade `grid-template-areas` permite dar nome a áreas específicas da grelha, facilitando o posicionamento dos itens de acordo com esses nomes. Isso torna o código mais legível e fácil de manter. Além disso, é possível ajustar o alinhamento e o espaçamento dos itens com propriedades como `justify-content`, `align-content` e `grid-gap`.

2.3 Referências para consulta

Nesta ficha irá praticar estas tecnologias. Sugere-se que utilize referências para saber mais sobre cada uma delas. Sobre HTML pode usar, por exemplo, o Capítulo 4 do HTML Living Standard ou o guia de referência HTML do W3Schools. Sobre CSS e CSS Grid, os guia de referência CSS do W3Schools ou da Mozilla.

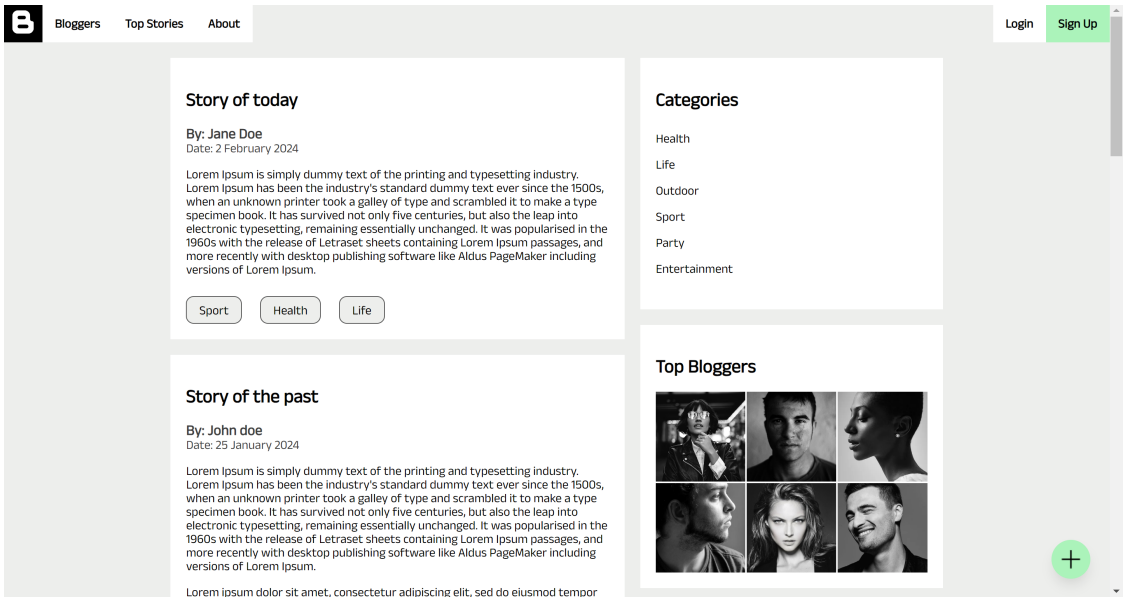

Figura 1: Website de blogs

3 Exercícios

Considere o website referente a uma rede de bloggers apresentado na Figura 1. O website é constituído pelos seguintes elementos:

- Uma *navigation bar* (no topo) que contém o logótipo, botões para navegar entre as páginas do website e botões que permitem a autenticação dos bloggers.
- Um conteúdo principal (*main*) em que são mostrados os últimos artigos publicados.
- Uma *sidebar* (à direita) que permite filtrar artigos por categoria.
- Uma segunda *sidebar* (por baixo das primeira) que identifica o top 6 dos bloggers registados.
- Um botão que permite criar um artigo (no canto inferior direito).
- Um rodapé (*footer*) que identifica o copyright (não visível na imagem).

Considere também a implementação base do website fornecida com esta ficha. Esta contém todos os ícones necessários para a construção do website de blogs, assim como as imagens referentes ao top 6 dos bloggers registados. Contém um ficheiro HTML que representa o esqueleto base da estrutura do website e um ficheiro

CSS que representa o *stylesheet* do website. Este ficheiro CSS contém a palete de cores do website.

Pretende-se, com esta ficha, que resolva os seguintes sete exercícios, que permitem chegar ao website apresentado na Figura 1.

3.1 Exercício 1

Comece por definir o *layout* base do website. Para isso utilize o CSS grid no elemento com o id "`app`" e explore o conceito das *grid template areas*.

O *layout* deverá ser constituído por 5 linhas e 4 colunas, sendo que nesta *grid* deverá constar uma área que deverá conter a *nav bar*, outra área para a *main* que conterá os artigos, uma área para a lista de categorias, uma área para o top 6 bloggers e, finalmente, uma área para o *footer*.

A primeira linha, que corresponde à *nav bar* deverá ter uma altura de 50 pixeis, a segunda e terceira linhas deverão ter o tamanho ajustável aos seus conteúdos, a quarta linha deverá ocupar o espaço disponível, e a última linha (correspondente ao *footer*) deverá ter 100 pixeis de altura.

Quanto às colunas: a primeira e última coluna deverão ocupar 1 fração da página do website, a segunda 3 frações e a terceira 2 frações. Este *layout* deverá ter a cor correspondente à variável `background-color` e deverá ter um espaçamento de 20 pixeis entre as suas linhas e colunas.

No final, aplique os nomes das áreas que definiu aos elementos `<nav>`, `<main>`, `<section>`, `<aside>` e `<footer>`. O resultado destas operações está ilustrado na Figura 2. Use as *dev tools* do seu browser para visualizar as propriedades que escreveu.

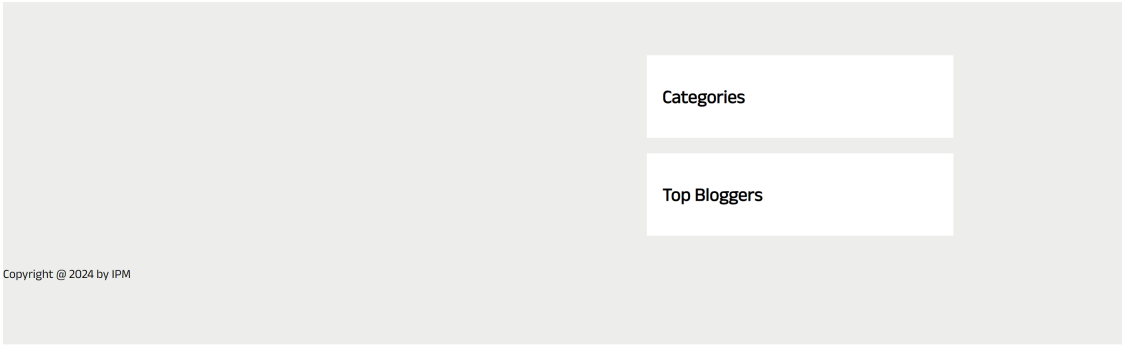

Figura 2: Resultado do *layout* base

3.2 Exercício 2

Finalize o *footer* do website. O *footer* deverá ter o texto centrado horizontalmente e verticalmente. O seu background deverá ser igual à variável `light-color`. Explore o CSS flexbox e as suas opções de alinhamento para esta questão. O resultado deverá ser semelhante ao da Figura 3.

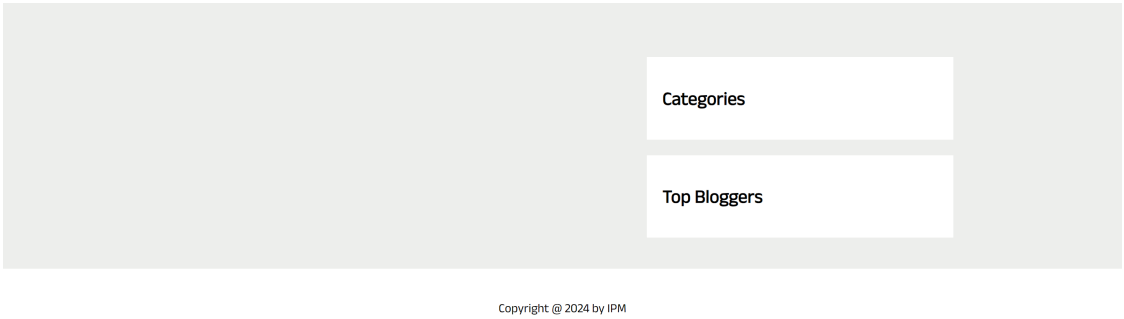

Figura 3: Resultado do footer

3.3 Exercício 3

Crie a *sidebar* da lista de categorias tal como apresentado na Figura 4.

1. Comece por criar a lista de categorias dentro da tag `<aside>` do ficheiro HTML. Use para isso as tags `<ul>` e `<li>`. A lista não deverá conter nenhum *padding* e os seus elementos deverão conter um *padding* vertical de 0.5rem e um cursor apropriado que indique ao utilizador que são clicáveis. Sugestão: atribua uma classe à lista, para facilitar a sua seleção no CSS.
2. Adicione mais *feedback* de utilização aos elementos da lista ao mudar a sua cor para a variável `accent-color-dark` sempre que o utilizador passar o rato sobre estes.

3.4 Exercício 4

Crie a lista de artigos. Esta lista deverá ser adicionada ao elemento `<main>` presente no ficheiro HTML descarregado. O processo deverá dar origem ao resultado apresentado na Figura 5. Para testar o *scrolling* do website adicione mais que um artigo:

1. Comece por criar um artigo a partir da tag `<article>`. Cada artigo deverá pertencer à class `card` (já disponibilizada no ficheiro CSS descarregado) e ter

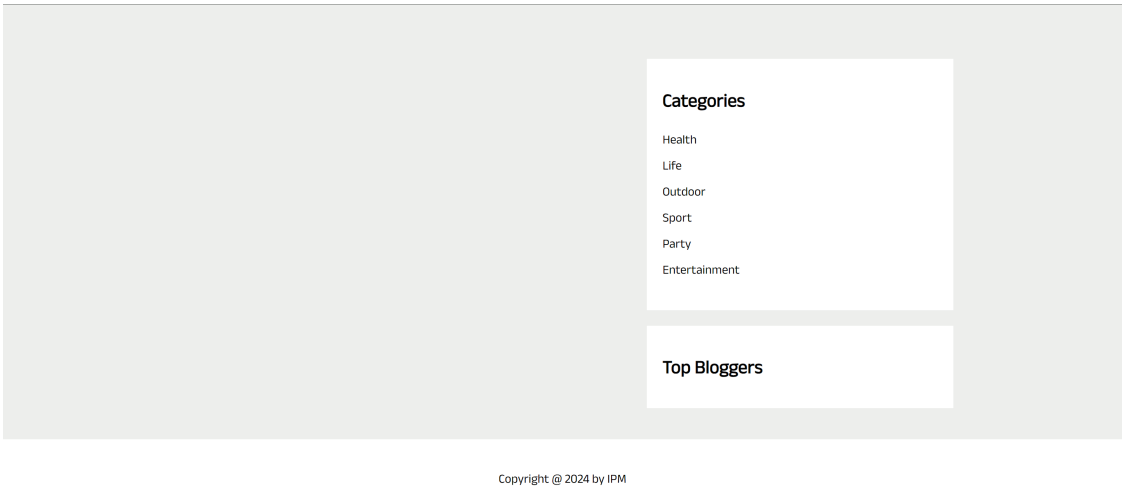

Figura 4: Resultado da sidebar de categorias

um `margin-bottom` de 20 pixeis. Cada artigo contém um título (através de uma tag `<h1>`), a indicação do nome do autor (através de uma tag `<h3>`), uma data (tag `<p>`) e um texto. O nome do autor e a data não deverão conter quaisquer margens e deverão ter a cor representada pela variável `text-secondary-color`.

2. Adicione uma lista de categorias no final do artigo. A lista deverá ser visível após o texto do artigo. Cada categoria possui um *padding* vertical de 0.5rem e um *padding* horizontal de 1.5rem. Possui uma margem à direita de 20 pixeis e ao topo de 10 pixeis. Além disso contém uma `border` de 1 pixel de cor igual à da variável `border-color` e um raio de 10 pixeis. Finalmente possui um fundo correspondente à variável `background-color`. Como fazer com que esta lista de categorias seja apresentada numa linha?

3.5 Exercício 5

Crie a *navigation bar* tal como ela é ilustrada na figura 1.

1. Comece por permitir à *nav bar* que o seu conteúdo seja mostrado na horizontal. Para isso use a *CSS Flexbox*.

   Crie o logótipo e os diferentes botões da *navigation bar* dentro da *tag* <nav> do ficheiro HTML. Para o logótipo use a imagem "logo.png". Esta imagem tem uma enorme resolução, sendo necessário redimensionar a mesma para 50 pixeis. Pode fazer isso diretamente no ficheiro HTML, ou no CSS.

   Crie os botões "Bloggers", "Top Stories", "About", "Login" e "Sign Up". Cada um destes botões possui o texto centrado verticalmente e horizontalmente, um

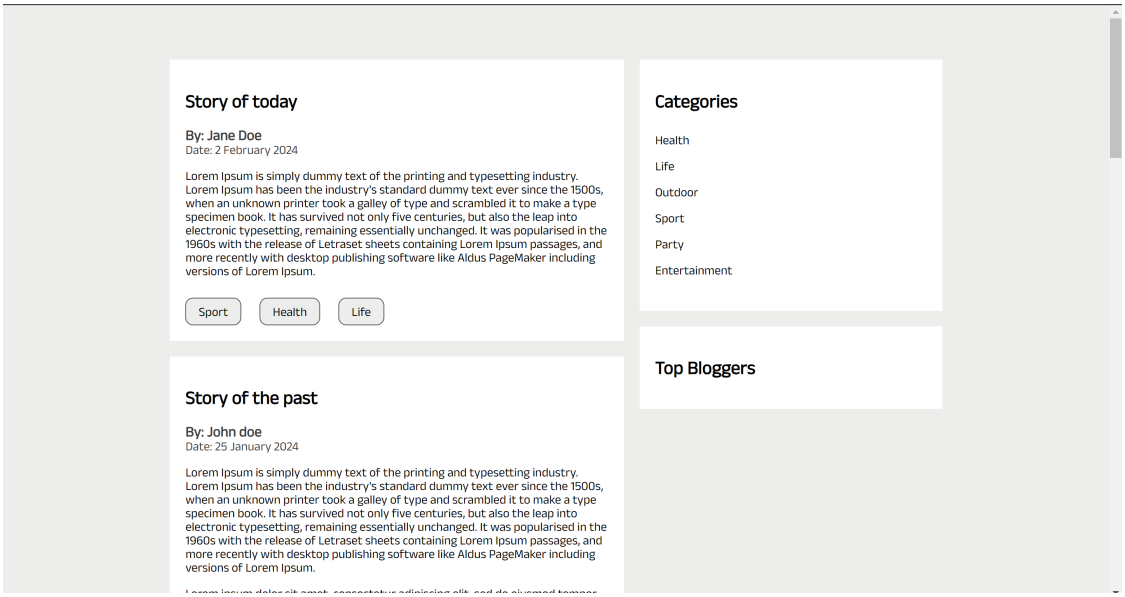

Figura 5: Resultado da lista de artigos

padding de 1rem, o fundo da cor definida na variável "light-color", uma *font-weight* de 600 e um cursor que indica ao utilizador que é clicável. Ao passar o rato por cima de um destes botões, o seu fundo deverá mudar para a cor "accent-color". O botão de "Sign Up" deve ter como cor de fundo "accent-color" e quando o utilizador move o rato por cima deste, a cor do seu texto deverá mudar para "accent-color" e o seu fundo para "dark-color".

2. Experimente a propriedade *justify-content: space-between* da *CSS flexbox* para distribuir os elementos na nav bar. O resultado é o esperado? Como alinhar apenas os botões de autenticação do lado direito?

3.6 Exercício 6

Considere os valores possíveis para o atributo de posicionamento de CSS (atributo position):

- *static* - O posicionamento padrão de CSS. Todos os elementos possuem este posicionamento por omissão, sendo posicionados de acordo com o fluxo normal do documento.
- *relative* - O elemento pode ser deslocado relativamente à sua posição normal.
- *absolute* - O elemento é posicionado relativamente ao elemento pai mais próximo na hierarquia que não seja *static*.

- *fixed* - O elemento é posicionado relativamente à janela do navegador. O elemento mantém-se fixo mesmo quando se efetua *scroll*.
- *sticky* - O elemento é posicionado de acordo com o fluxo normal do documento, mas comporta-se como *fixed* quando a sua posição está dentro de um limite especificado.

Qualquer tipo de posicionamento não estático utiliza os valores das propriedades *top, left, right* e *bottom* para calcular a posição do elemento.

Dado o atributo de posicionamento mencionado, crie o botão que permite a um utilizador criar um artigo, tal como mostrado na Figura 1. Este botão deverá permanecer fixo, mesmo que o utilizador faça *scroll*, deverá conter um ícone (na ficha é fornecido o ícone "plus.png"), ter como fundo a variável "accent-color", uma altura e largura de 50 pixeis e um raio de 25 pixeis. Deverá ainda ter um distanciamento do canto inferior direito do website de 25 pixeis. Finalmente, deverá ter um cursor apropriado e uma sombra para dar mais informação de relevo ao utilizador. Sugestão: explore a página Beautiful CSS box-shadow examples, do website CSS Scan, que contém vários exemplos de sombras a aplicar.

3.7 Exercício 7

Crie a galeria de imagens que corresponde aos 6 melhores bloggers.

1. Comece por acrescentar ao elemento <section> uma *CSS grid* que deverá conter 2 linhas e 3 colunas. Cada um destes elementos deverá ocupar uma fração, ter um espaçamento de 2 pixeis e centrar os seus elementos. Use o método *repeat* para a construção da *grid*. As imagens relativas aos bloggers estão disponíveis com a ficha e deverão ter uma largura de "100%" e uma altura máxima de "100%".
2. Acrescente um *overlay* em cada imagem referente a um blogger tal como apresentado na Figura 6. O *overlay* deverá mostrar o nome do blogger e ter um *padding* vertical de 0.5rem. A cor do seu texto deverá ser igual à variável "light-color" e o seu fundo deverá ser preto e semi-opaco (use o método *rgba*). Finalmente, este *overlay* apenas deverá ser mostrado quando o utilizador move o cursor por cima da imagem do blogger. O que deverá mudar na estrutura da galeria de bloggers para que isto seja possível? Use os posicionamentos *relative* e *absolute*. Sugestão: investigue como pode controlar a visibilidade dos elementos, utilizando CSS.

Figura 6: *Overlay* da galeria de bloggers