HTML (II)

Linguagens de Anotação de Documentos

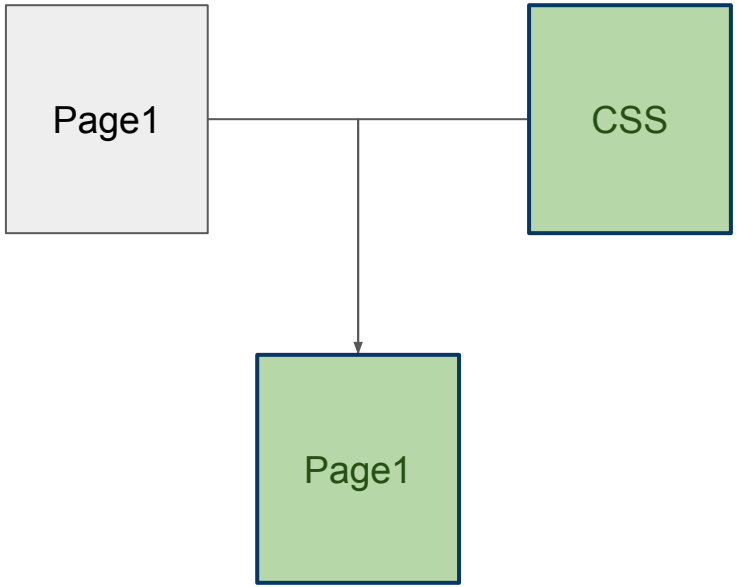
Cascading Style Sheets

- O HTML foi desenvolvido para definir o *conteúdo* das páginas
- Há algumas anotações que definem *estilo*, mas causam problemas aos *designers*
    - Algumas foram removidas em versões mais recentes
- O CSS é uma linguagem para escrever *folhas de estilo*
- Resolve este problema, separando a estrutura do documento da sua formatação

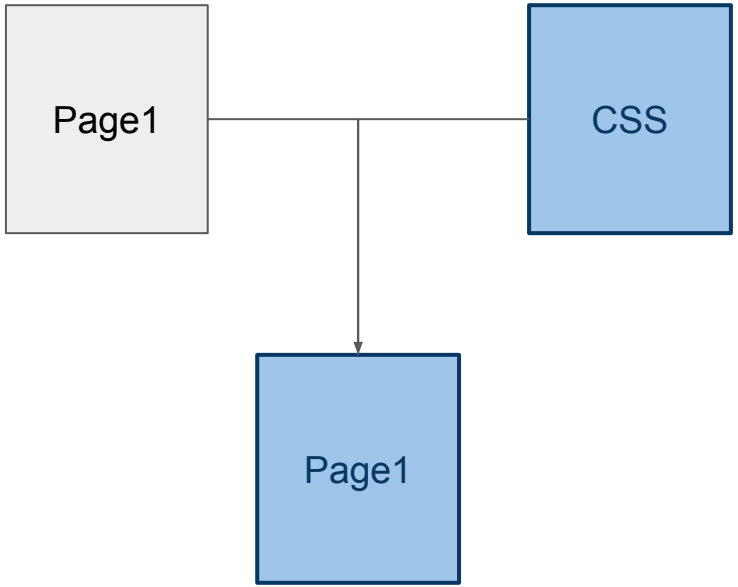
Cascading Style Sheets

- Em HTML definimos o conteúdo usando as anotações estruturais
- Em CSS definimos como cada uma dessas anotações deve ser apresentada

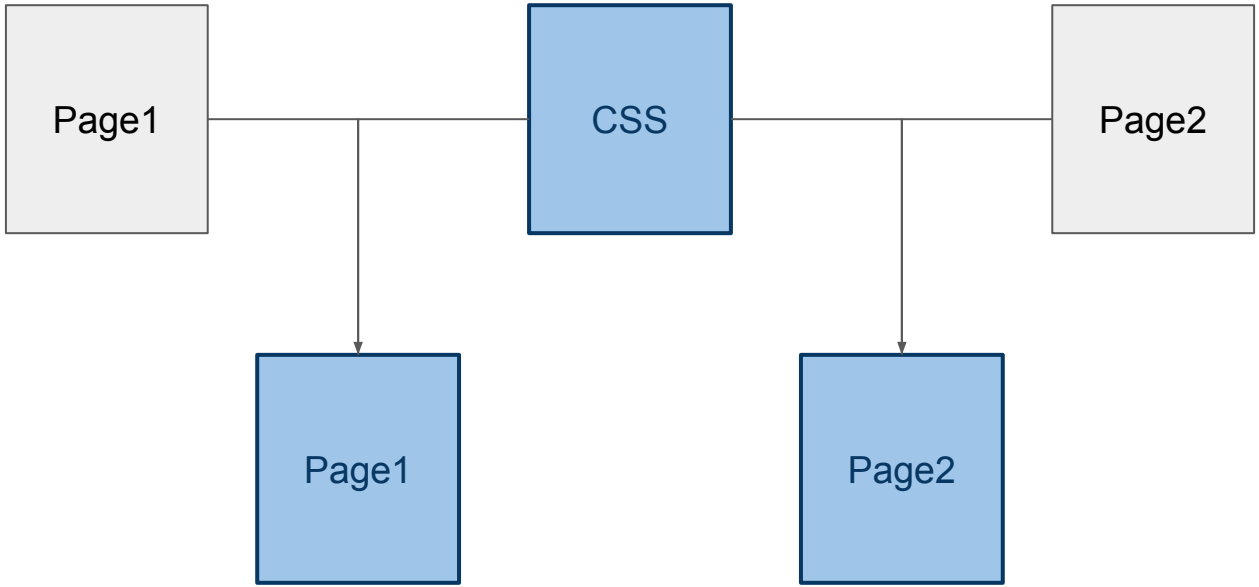
Cascading Style Sheets



Cascading Style Sheets



Cascading Style Sheets



Sintaxe CSS

- Sintaxe das regras de estilo:
  - Seletor
  - Declarações: propriedades + valores

```
seletor { propriedade: valor; …; propriedade: valor; }
```

Sintaxe CSS

- Exemplo: alterar a formatação de todos os títulos `h1`

```
h1 {
    color: red;
    font-size: 14px;
    border: 1px solid;
}
```

Sintaxe CSS

- Os *seletores* definem todos os elementos HTML que serão afetados pela regra
  - Tipos de anotações
  - Identificadores de elementos (os mesmos usados nas referências cruzadas)
  - Classes

Sintaxe CSS

- As *propriedades* definem o estilo de todos os elementos HTML identificados pelos seletores
  - Cores
  - Dimensões
  - *Borders*
  - …

Sintaxe CSS

- *Propriedades* CSS típicas
  - `color`
  - `background-color`
  - `font-size`
  - `font-family`
  - `text-align`
  - `border`
  - `margin`
  - `padding`

Definir CSS

- Vários mecanismos para definir o estilo com CSS:
  - Folha de estilo externa
  - Folha de estilo interna
  - Estilos “*inline*”
- Estilos “*inline*” devem ser usados para formatações específicas; folhas de estilo para tudo o que deva afetar múltiplos elementos

Definir CSS

- Folhas de estilo são definidos no cabeçalho do documento HTML (`head`)
- Externas, com `link` (extensão *css*):

```
<link rel="stylesheet" type="text/css" href="mystyle.css">
```

- Internas, elemento `style`:

```
<style> definições_css </style>
```

Definir CSS

- Estilos inline são atribuídos diretamente aos elementos com o atributo `style`

```
<h1 style="color: blue;border: 1px;"> título </h1>
```

- Devem ser apenas usados se a regra não se aplicar a mais elementos

Tutorial

https://www.w3schools.com/css/

Imagens

- *Imagens* com `img`
- Elemento HTML sem conteúdo (as imagens são ficheiros externos)

```
<img src="endereço_imagem">
```

- Endereço pode apontar para imagens locais ou remotas
- Ligações em imagens: simplesmente aninhar

```
<a href=...> <img src=...> </a>
```

Imagens

- Estilo nas imagens
  - `width`, `height` para dimensões
  - `float` para alinhamento
  - `display` para converter em bloco

Listas

- Listas *ordenadas* (`ol`) ou *não-ordenadas* (`ul`)
- Elementos da lista com `li`
- *Descrições* (`dl`) definem termos (`dt`) com a sua descrição (`dd`)
- Podem conter outros elementos aninhados
- Propriedades CSS definem o estilo (e.g., `list-style-type`)

Listas

```
<ul style="list-style-type:circle">
    <li>item 1</li>
    <li>item 2
        <ul>
            <li>item 2.1</li>
            <li>item 2.2</li>
        </ul>
    </li>
    <li>item 3</li>
    <li>item 4</li>
</ul>
```

Tabelas

- *Tabelas* definidas por `table`
- Linhas com `tr` e colunas (células) com `td`
- Cabeçalhos podem ser distinguidos com `th`
- Propriedades CSS definem o estilo

Tabelas

```
<table style="width:100%">
    <tr>
        <th>Nome</th> <th>Número</th>
    </tr>
    <tr>
        <td>João</td> <td>88128</td>
    </tr>
    <tr>
        <td>Joana</td> <td>892323</td>
    </tr>
</table>
```

Tabelas

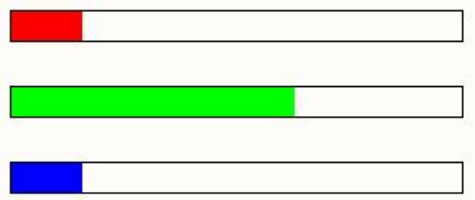
```
table, th, td {
    border: 1px solid black;
}

th {
    background-color: green;
    color: white;
    padding: 15px;
    text-align: center;
}
```

Cores

- 140 *cores* são suportadas em CSS diretamente pelo seu nome
- Para cores adicionais o mais prático é usar valores RGB (*red*, *green*, *blue*)
- Para cada canal, atribui-se um valor entre 0 e 255, o que dá cerca de 16 milhões de cores diferentes

```
color:rgb(40, 160, 40)
```



Blocos

- Cada elemento tem uma propriedade `display` que define se
  - É um `block`, ocupa toda a largura da linha (`h1`, `p`)
  - É `inline`, ocorrendo no meio do texto (`a`, `img`)
- Pode ser mudado no estilo
- Há elementos genéricos, sem significado intrínseco, dos dois tipos (`div` e `span`, respectivamente)
- São usados para agrupar outros elementos, para facilmente alterar propriedades de vários elementos

Meta-dados

- Já vimos que no cabeçalho `head` definimos meta-dados como o `title` e o `style`
- Elementos `meta` definem outros meta-dados relevantes, usados pelos *browsers* e pelos motores de pesquisa
  - `charset`
  - `description`
  - `keywords`

HTML5

- O HTML5 lançado em 2014 modernizou o HTML
- Funcionalidades que necessitavam de tecnologias adicionais (e.g., *Flash*) são agora *standard*
- Novos elementos, e.g., estruturais com diferente semântica (`article` ou `section`) ou multimedia (`video` e `audio`)
- Mais funcionalidades para portabilidade (i.e., diferentes dispositivos)
- Já é suportado por todos os *browsers*

JavaScript

- JavaScript é a terceira tecnologia fundamental para a escrita do *front-end* de páginas *web*
- Linguagem de *programação*: permite definir páginas com conteúdo dinâmico
- Por exemplo, criar novos elementos, ou alterar o seu conteúdo, de acordo com acções do utilizador

```
document.getElementById("id").innerHTML = "Olá Mundo!";
```

Take-home Lesson

- O HTML e o CSS são as peças básicas do *front-end* de todas as páginas *web* (i.e., aquilo que acontece do lado do *cliente*)
- A separação de preocupações disponibilizada pelo HTML/CSS é fundamental para a gestão de páginas de grande dimensão e complexidade
- São linguagens simples, que permitem escrever (ou modificar) páginas facilmente e rapidamente
- Mesmo que se usem ferramentas para a geração de páginas automáticas, modificações personalizadas acabam sempre na edição de HTML/CSS