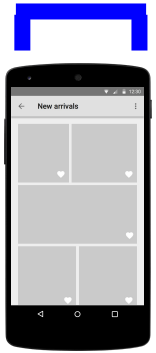
Cascading Style Sheets

Sites Responsivos

Media Queries

O que é uma media query? (media1.html)

400px




```
@media (max-width: 400px) {
    body {
        background-color: red;
    }
}
```

As *media queries* são recursos pelos quais podemos definir layouts diferentes de acordo com o tamanho da viewport do navegador.

900px


```
@media (min-width: 900px) {
    body {
        background-color: blue;
    }
}
```

Regras da media query (media2.html)

Existem várias regras que podem ser utilizadas nas condições da media query, mas na prática o uso mais comum é da propriedade *min-width*, com desenvolvimento de layouts mobile-first.

`@media (max-width: 500px)` - viewports com no máximo 500 pixels de comprimento

`@media (width: 600px)` - viewports com exatamente 600 pixels de comprimento

`@media (min-width: 900px)` - viewports com no mínimo 900 pixels de comprimento

`@media (max-height: 300px)` - viewports com no máximo 300 pixels de altura

`@media (height: 350px)` - viewports com exatamente 450 pixels de altura

`@media (min-height: 400px)` - viewports com no mínimo 500 pixels de altura

Regras da media query (media3.html)

Outros exemplos:

@media (device-width: 1366px) - dispositivo com 1366px de comprimento

@media (max-device-width: 2000px) - dispositivo com no máximo 2000px de width

@media (min-device-width: 500px) - dispositivo com no mínimo 500px de width

@media (device-height: 400px) - dispositivo com 400px de altura

@media (max-device-height: 800px) - dispositivo com no máximo 800px de altura

@media (min-device-height: 300px) - dispositivo com no mínimo 300px de altura

@media (orientation: portrait) - orientação que o height é maior que o width

@media (orientation: landscape) - orientação que o width é maior que o height

Media Types (media4.html)

Além das condições, podemos especificar o tipo de média que queremos aplicar a estilização. Isso é feito através das media types.

@media all - Aplicado a todas as medias. Este é o padrão.

@media screen - Aplicado aos dispositivos com telas.

@media print - Aplicado às impressoras. Útil para adaptar o estilo e limpar a página para melhorar a impressão do documento.

Existem várias outras media types, sendo que a maioria delas foi descontinuada pela W3C. Há também a media *speech*, que apesar de ter sido mantida na especificação da W3C, os navegadores atuais não suportam.

Exemplos de uso (media5.html)

```
<link rel="stylesheet" href="css/mobile.css" media="(max-width:600px)" />
```

É possível definir a *media* diretamente na importação do CSS externo.

```
@media not print
```

NOT: Utilizada para definir a negação de uma condição. Exemplo: todos, exceto o *print*

```
@media (min-width: 550px) and (max-width: 700px)
```

AND: Só aplica o estilo se todas as condições forem verdadeiras.

```
@media (max-width: 600px), (min-width: 700px)
```

OR: Aplica o estilo se pelo menos uma das condições for verdadeira.

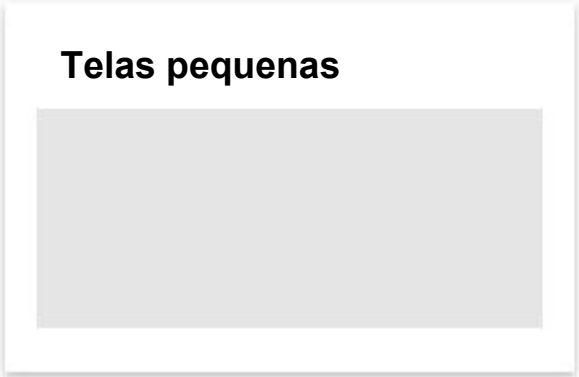
```
@media only screen and (min-width: 510px)
```

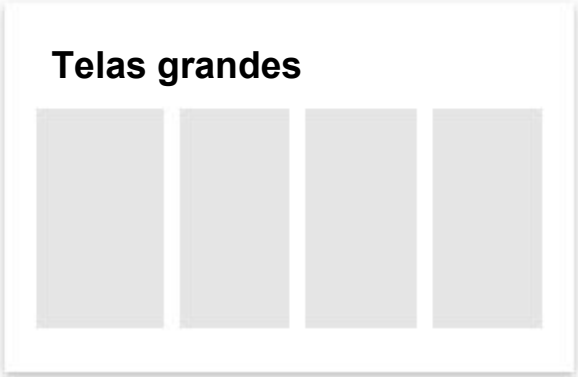
ONLY: Evita que browsers antigos apliquem o estilo sem atender às condições.

Layout Fluido

Uso prático das media queries (fluido1.html)

Através das media queries é possível construir layouts adaptáveis ao tamanho da viewport.





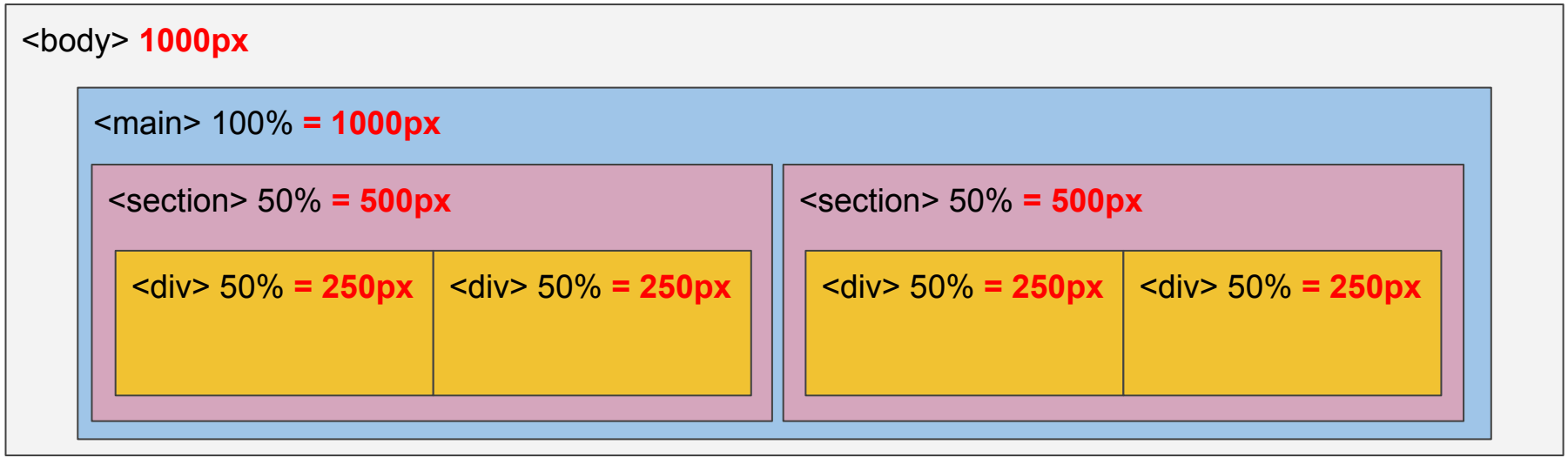

```
section {
    float: left;
    width: 100%;
}
```

```
@media (min-width: 500px) {
    section {
        width: 50%;
    }
}
```

```
@media (min-width: 900px) {
    section {
        width: 25%;
    }
}
```

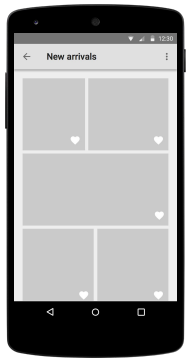
Uso prático das media queries (fluido2.html)

Os elementos internos também ter seus tamanhos definidos com %, sendo que neste caso, o percentual é relativo ao componente pai.



O alinhamento dos elementos internos podem ser realizados via *float: left* ou *display: inline-block*, sendo que este último pode ter pequenos espaçamentos entre os elementos devido à natureza padrão de texto no HTML. Veja no arquivo *fluido2.html* um “macete” para resolver este problema.

Limites do layout (fluido3.html)



320px **1200px**

Crie limites para sua aplicação com as propriedades *min-width* e *max-width*. Existem aparelhos iPhone com 320 pixels horizontais. Ao limitar este tamanho mínimo, no caso onde o usuário reduza para um comprimento inferior, o site vai reduzindo de tamanho para não “quebrar” o layout. Defina também um tamanho máximo, sempre lembrando de centralizar o conteúdo para o caso de acessos em monitores/TVs maiores.
Em termos práticos, isso pode ser feito em uma tag geral (<main>, por exemplo), limitando-a com tamanho mínimo e máximo.

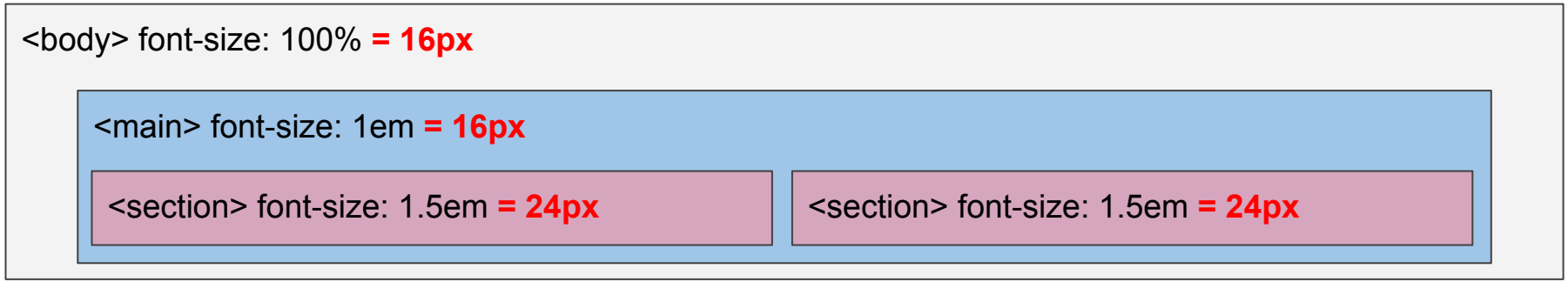
Centralizando o layout (fluido4.html)

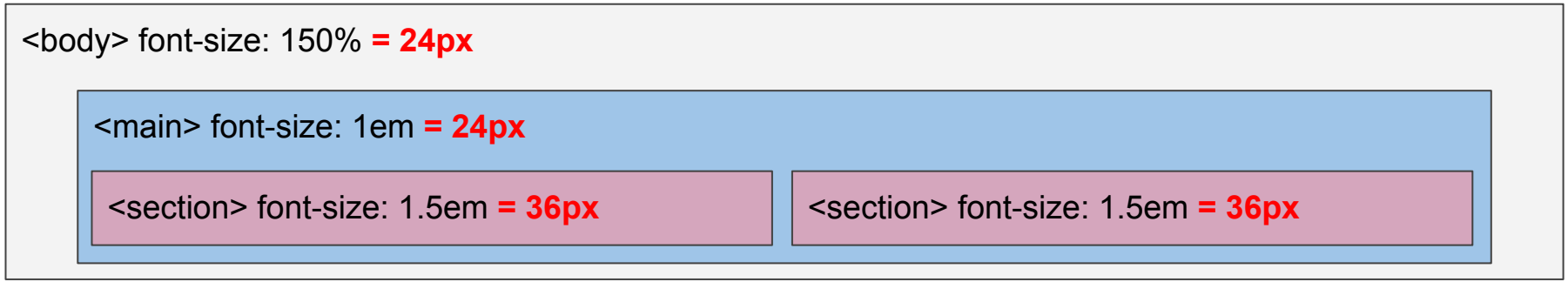
Além dos limites do layout, você pode centralizá-lo com um simples *margin: 0 auto;*
Dessa forma ele vai se adaptar melhor em telas muito grandes.


```
main {
    min-width: 320px;
    max-width: 1000px;
    margin: 0 auto;
}
```

Fontes e espaçamentos relativos (fluido5.html)

Ao definir fontes com tamanho relativo (% ou *em*), é possível alterá-las a partir de um pequeno ajuste na tag <body>. O uso de medidas relativas (*em*) no *padding*, *border,* e outras propriedades associadas a textos, é uma boa prática.





Dicas gerais

- Há exceções, mas como regra geral, **não** retire conteúdo dos usuários mobile. Tudo que existir na versão desktop deve estar disponível na versão mobile.
- Sempre use:
  `<meta name="viewport" content="width=device-width, initial-scale=1.0">`
- Sempre use:
  `<link rel="icon" href="favicon.ico">`
- **Não** desabilite o zoom do usuário (dois cliques na tela):
  `user-scalable=no`
- Podemos retirar elementos da versão desktop usando:
  `display: none;`

Dicas gerais

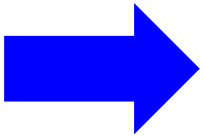
Sempre faça seu design pensando em:

- **Mobile-first:** crie seu layout primeiro para o celular. Depois ajuste para o desktop.
- **Conexões 3G ou Offline-first:** a página deve ser criada para suportar conexões muito lentas ou até mesmo adotar recursos onde não exista conexão com a Internet (Progressive Web Apps)
- **Touch-first:**
  - **Tapping:** um toque simples
  - **Double-tapping:** dois toques simples
  - **Swiping:** movimento horizontal ou vertical com o dedo
  - **Flinging:** semelhante ao Swiping, porém muito rápido
  - **Long Pressing:** toque longo
  - **Pinch-Open:** dois dedos se afastando (zoom-in)
  - **Pinch-Closed:** dois dedos se juntando (zoom-out)

Botões mobile (fluido6.html)

Estudos indicam o tamanho ideal para os botões mobile.

MIT: 12.4 a 15mm ⇒ 47 a 57 pixels
Microsoft: 9mm ⇒ 34 pixels
Apple: 6.8mm ⇒ 26 pixels
Mozilla: 5.9 a 9mm ⇒ 22 a 34 pixels
Google: 10.6 a 14.8mm ⇒ 40 a 56 pixels



Mais utilizados:
7 a 9mm
26 a 34 pixels

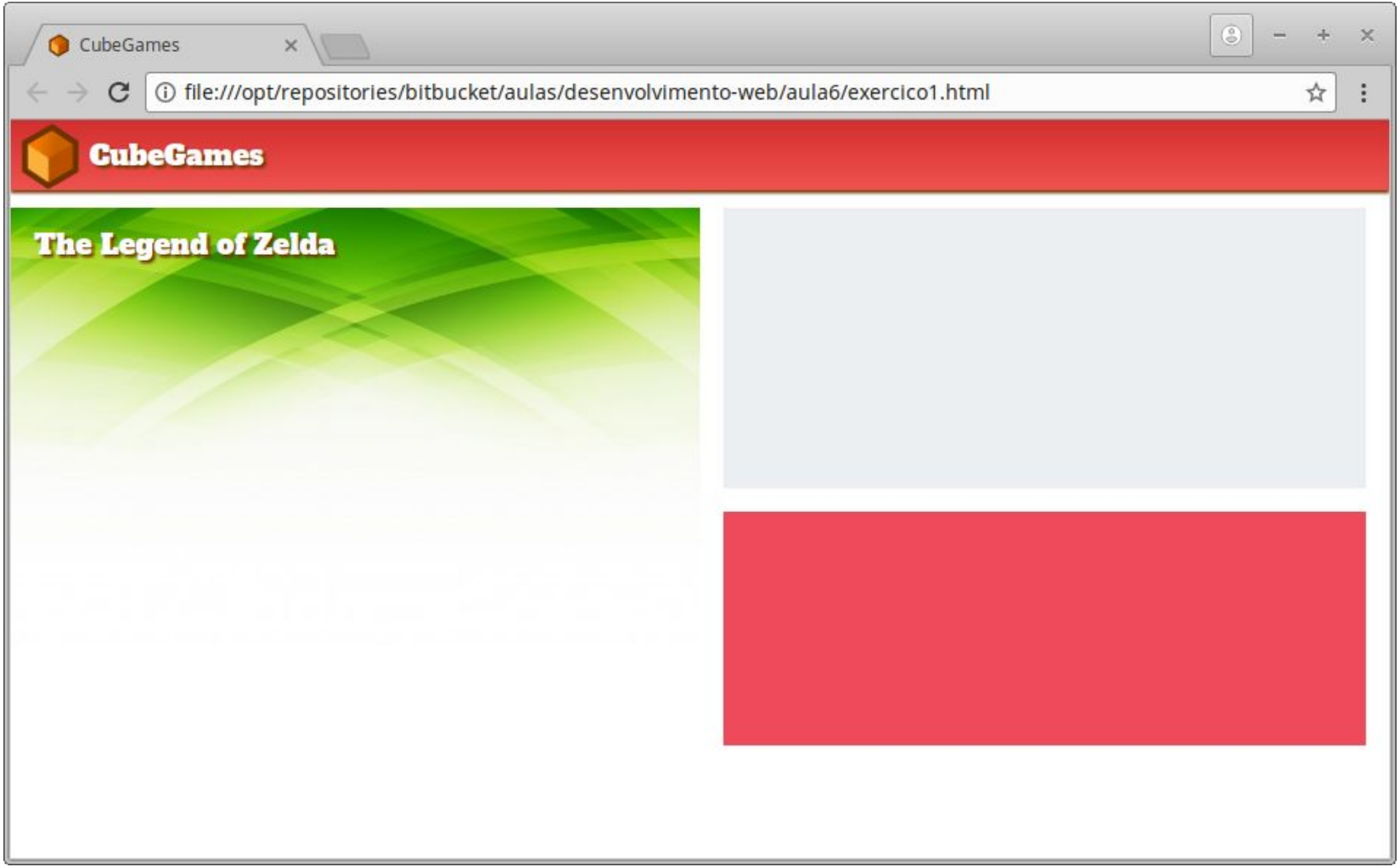
Espaçamento mínimo: 10 pixels entre os botões
Cores: botões primários devem ter cores mais fortes, enquanto que botões secundários devem ter cores mais suaves
Affordance: no caso do botão, é a capacidade de informar que aquele elemento é um item *clicável* (formato, cores, sombras, etc).
Floating Action Button: devem ser retirados da versão desktop.

Exercício 1

Crie uma página com:

- Um cabeçalho fixo com o seguinte comportamento:
  - Título centralizado, **sem** exibir o logo do site
  - Para páginas com mais de 600px, o título deve ser alinhado a esquerda e o logo deve ser exibido (também à esquerda)
- Três seções com 100% de comprimento:
  - A primeira com uma imagem de background e um <h1> estilizado.
  - A segunda com um fundo cinza-claro.
  - A terceira com um fundo vermelho.
- Para páginas com mais de 800px, as seções devem ter *width* de 50%

Observação: como as seções não possuem conteúdo, por enquanto defina um *height* adequado para que elas sejam exibidas corretamente.


CubeGames
file:///opt/repositories/bitbucket/aulas/desenvolvimento-web/aula6/exercico1.html
CubeGames
The Legend of Zelda

Imagens

Regra simples para imagens (imagem1.html)

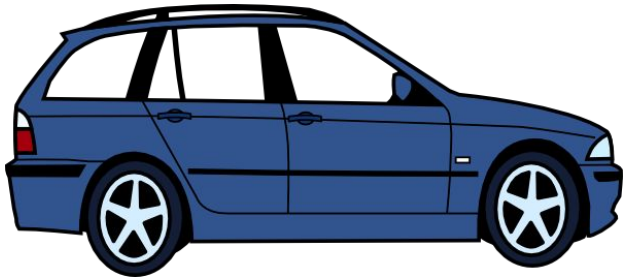
Imagens são representadas em pixels. Neste caso, definir um *width* de 100% pode “esticar” a imagem até que a mesma perca qualidade. Uma técnica muito simples e bastante utilizada é definir a seguinte sintaxe:

```
img {
    max-width: 100%;
}
```

Imagens vetoriais (imagem2.html)

Imagens SVG são renderizadas pelo navegador e podem ter seu tamanho ajustado **sem** piorar a qualidade. Neste caso, podemos utilizar *width* com 100% sem prejudicar o layout.

```
img {
    width: 100%;
}
```



O problema é que o IE8 (ou inferiores) não suportam. O código para contornar este problema é o seguinte.

```
<img src="img/car2.svg" alt="Carro"
    onerror="this.src='img/car2.png'; this.onerror=null" />
```

Reduzir qualidade das imagens (imagem3.html)

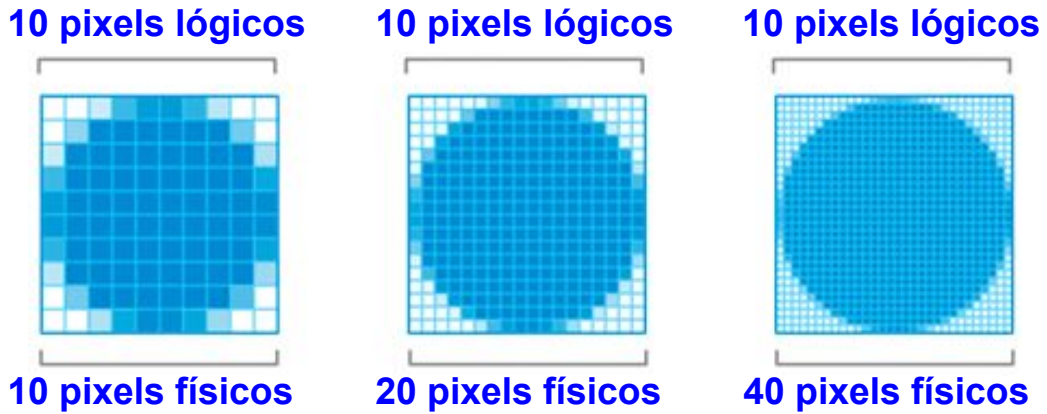
Quando se trata de imagens na web, o ótimo é inimigo do bom. Podemos reduzir o tamanho do arquivo (em KB) da imagem com softwares especializados ou através de sites como:
https://www.reduceimages.com/
http://compressjpeg.com/
http://optimizilla.com/
http://www.imagesmaller.com/

Qualquer redução na qualidade já é suficiente para diminuir a imagem, mas há estudos que indicam que reduzir a qualidade para 20% pode não impactar na visualização no navegador e é capaz de diminuir o arquivo em mais de 80%.

Telas de alta resolução (imagem4.html)

Pixel Lógico (CSS) ≠ Pixel Físico (Telas)

As telas de alta resolução têm vários pixels físicos por pixel lógico. Este termo também é conhecido por Device Independent Pixel (ou DIP, definição do Android), Pontos (definição da Apple) e Pixel CSS (definição da W3C).
Um único **pixel lógico** pode corresponder diretamente a um único **pixel físico** ou a vários **pixels físicos**. Quantos mais **pixels físicos**, maiores os detalhes do conteúdo exibido na tela.



Telas de alta resolução (imagem4.html)

DPI (Dots Per Inch) = Pontos por Polegada
dPR (Device Pixel Ratio) = Relação entre pixels físicos e pixels lógicos

Telas com alto DPI (2 ou mais pixels lógicos para 1 pixel físico) é possível que sua imagem fique "pixelada", pois uma imagem de 100px será exibida em 200px.

1 DPI

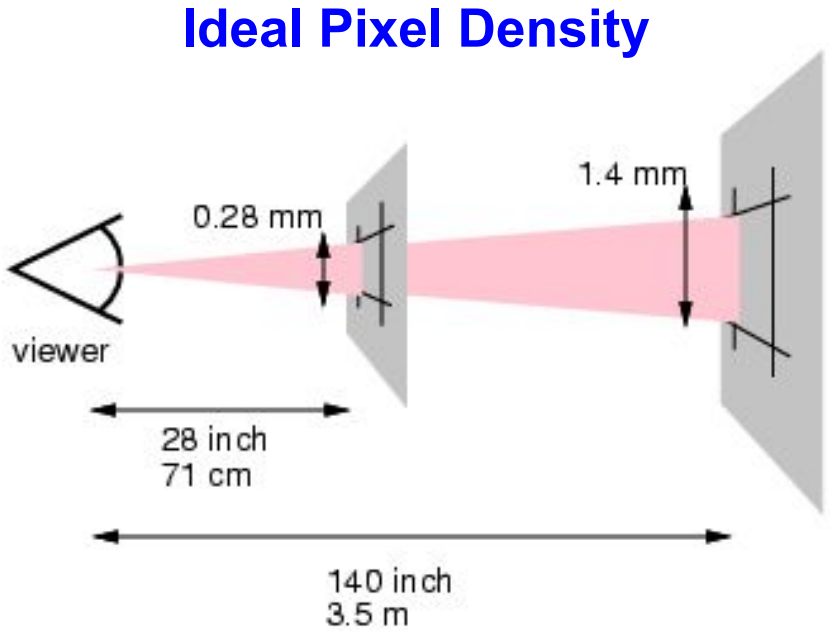
2 DPI

3 DPI

Telas de alta resolução (imagem4.html)

Exemplos de telas:
Telas Desktop (média): 96dpi ⇒ 1 dRP
Telas Mobile (média): 150dpi ⇒ 1.5 dRP
iPhone 4: 326dpi ⇒ 2 dRP
Galaxy S5: 432dpi ⇒ 3 dRP

Outros: http://dpi.lv/



Blz, mas e dai? (imagem5.html)

dppx (dots per pixel)

Sabendo disso, é possível especificar tamanhos diferentes para o background de acordo com dRP do dispositivo. Deve-se utilizar a condição *min-resolution*

```
a.link {
    background-image: url(img/dpi/imagem_1x.png);
}
@media (min-resolution: 2dppx) {
    a.link {
        background-image: url(img/dpi/imagem_2x.png);
    }
}
```

E as tags <img>? (imagem6.html)

- **Via JavaScript:** Obtém o DPI do dispositivo e altera o conteúdo de <img>. **Atenção:** usar com muito cuidado, pois isso obrigará o navegador a fazer 2 downloads por imagem. Só usar para caso muito especiais.
- **Atributo *srcset*:** Em navegadores novos já existe o atributo *srcset* que permite definir as imagens e a taxa DPI. **Atenção:** sempre defina o *src* também, para suportar navegadores antigos.

```
<img src="img/dpi/imagem_1x.png" srcset="img/dpi/imagem_2x.png 2x" />
```

- **Nova tag *<picture>*:** Esta nova tag permite especificar os arquivos por media query. **Atenção:** o uso conjunto com <img> é obrigatório.

```
<picture>
    <source media="(min-resolution: 2dppx)" srcset="img/dpi/imagem_2x.png">
    <source srcset="img/dpi/imagem_1x.png">
    <img src="img/dpi/imagem_1x.png" />
</picture>
```

Exercício 2

Ajuste a página do Exercício 1 com:

- Define uma imagem responsiva para as seções. No caso das seções 2 e 3, coloque as imagens alinhadas no centro.
- Defina botões fixos nas seções 1 e 3. Na seção 2, crie um botão estilizado centralizado no centro.

Observação: como as seções agora terão a altura de suas imagens internas, pode retirar o atributo *height*.

CubeGames

file:///opt/repositories/bitbucket/aulas/desenvolvimento-web/aula6/exercicio2.html

CubeGames

The Legend of Zelda

Zelda é, na série de jogos eletrônicos The Legend of Zelda, da Nintendo, o nome dado às mulheres da família real de Hyrule, que têm o título princesa Zelda em jogos cronologicamente posteriores a Skyward Sword.

Promoção de Jogos PS4

Menu Responsivo

Menu responsivo - PASSO 1 (menu1.html)

Primeiramente coloque a imagem do *hamburger* posicionado da esquerda do header.

Aplique uma media query que esconda o menu para dispositivos com mais de 600 pixels de comprimento.

```
@media screen and (min-width: 600px) {
    a.menu {
        display: none;
    }
}
```

Menu responsivo - PASSO 2 (menu2.html)

Crie uma <nav> com links sobre os demais conteúdos (z-index) com um botão de fechar no topo esquerdo e links posicionados um sobre o outro. Ela deve ter posicionamento fixo, comprimento de 80% e altura de 100%.

```
<nav>
    <a id="fechar" class="menu" href="#">
        <img src="img/fechar.png" alt="Fechar menu" />
    </a>
    <a class="link">PlayStation 4</a>
    <a class="link">Xbox One</a>
    <a class="link">Nintendo Switch</a>
</nav>
```

Menu responsivo - PASSO 3 (menu3.html)

Para os dispositivos com no mínimo 600px, ajuste o <nav> para ficar fixo no cabeçalho, posicionando os links à direita.

Menu responsivo - PASSO 4 (menu4.html)

1. Agora vamos esconder a <nav> (para dispositivos menores que 600px) posicionando-a com *left: -80%.* Dessa forma, a <nav> ficará escondida à esquerda da viewport.
2. Crie uma classe menu-aberto que será aplicada ao <html> da página via JavaScript. Como o <nav> está “dentro” do <html>, vamos criar um estilo descendente para definir <nav> com *left: 0*, tornando-a visível.

   ```
   .menu-aberto nav { left: 0; }
   ```
3. Por fim, importe o código JavaScript no final do <body>.

   ```
   <script src="js/menu.js"></script>
   ```

Menu responsivo - PASSO 5 (menu5.html)

1. Acrescente uma transition de 500 milissegundos à propriedade *left* da tag <nav>.

```
transition: left 500ms;
```

2. Acrescente um pseudo-elemento after à classe menu-aberto ocupando toda a área da viewport e cor preta com transparência. Isso deixará o resto da página escurecida quando o menu for aberto.

```
.menu-aberto::after {
    /* estilos do css*/
}
```

Exercício 3

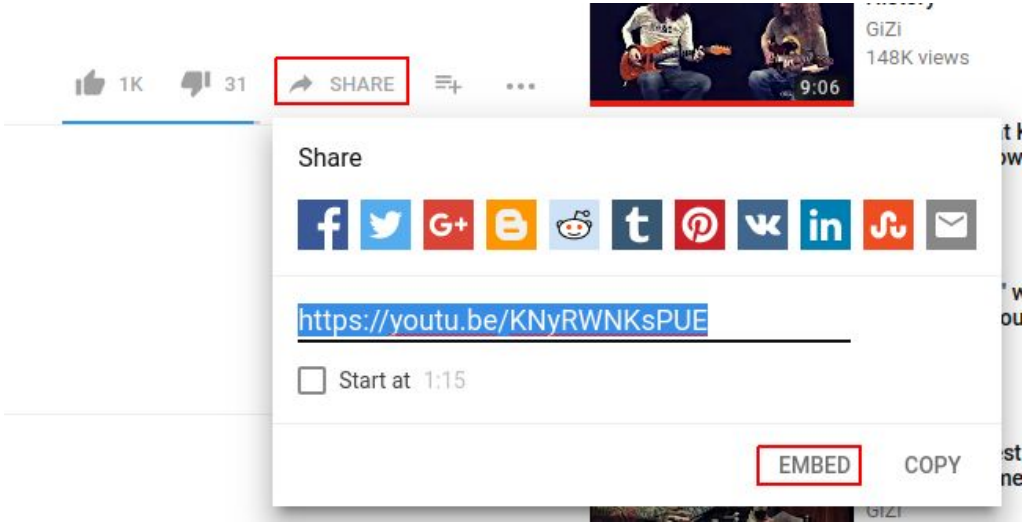
Ajuste a página do Exercício 2 com um menu responsivo que:

- Ocupe 80% dos dispositivos mobile, aparecendo à esquerda sobre o conteúdo do site. Este menu deve ser aberto e fechado via JavaScript e deve escurecer o conteúdo da página quando estiver aberto.
- Se adapte aos dispositivos maiores que 600 pixels fixando-se à direita do <header>.

Extra: vídeo responsivo

Vídeo responsivo (video.html)

O próprio YouTube oferece o código necessário para acrescentar uma tag de vídeo às suas páginas. Isso é possível pelo link **Embed**



Este link, porém, define um tamanho fixo para o vídeo em pixels. Veja como tornar a tag <iframe> responsiva no código video.html

Exercício 4

Acrescente um vídeo no código Exercício 3, mais especificamente à seção 1. Este vídeo deve ser responsivo.

Obrigado!