CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS VII - UNIDADE TIMÓTEO
Curso: Engenharia de Computação
Disciplina: Desenvolvimento de Sistemas
Professor: Odilon Corrêa da Silva

GUIA 03 - CSS: Layout Responsivo

Objetivo

- Criar layouts responsivos

Layout Responsivo

As páginas web são acessadas em diversos dispositivos, como computadores, celulares, tablets e TVs, cada um com diferentes tamanhos e resoluções de tela. Para garantir uma experiência consistente e agradável, é essencial que as páginas se ajustem automaticamente a essas variações, tornando-se responsivas.

Um layout responsivo busca adaptar seu conteúdo ao tamanho da tela do dispositivo utilizado, proporcionando uma melhor experiência ao usuário, independentemente do meio de acesso (Figura 01).

Figura 01: Layout responsivo

Entre os recursos do CSS para implementar essa adaptação, destacam-se as **media queries**, que permitem aplicar estilos específicos com base nas características do dispositivo, como largura e altura da tela. Elas são definidas usando **media types** e **media features**:

- **Media Type**: Define o tipo de mídia para o qual os estilos serão aplicados. Exemplos:
  - screen: Para monitores e telas de dispositivos
  - print: Para impressoras
  - all: Aplica-se a todos os tipos de mídia
- **Media Features**: Permitem definir condições baseadas nas propriedades do dispositivo, como **max-width**, **min-width**, orientation, entre outras.

Exemplo de media query:

```
@media screen and (max-width: 768px) {
    body {
        background-color: orange;
    }
}
```

No exemplo acima, a cor de fundo será alterada quando a largura da tela for menor que **768px** (`max-width: 768px`).

Exemplo

- A Figura 02 ilustra um layout responsivo utilizando media queries e CSS Grid Layout. Esse layout representa a abstração da estrutura básica página de acesso do Instagram.

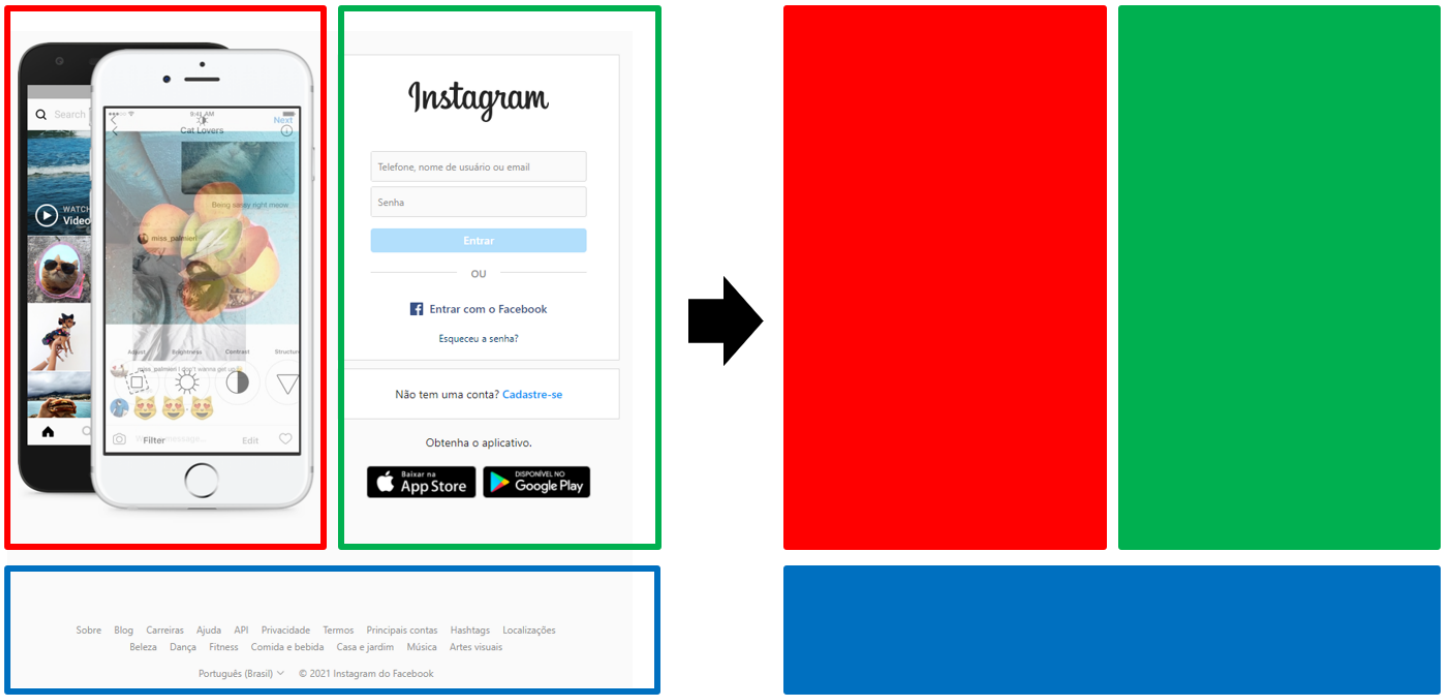

Figura 02: Layout do Instagram

```html
<html>
    <head>
        <meta charset="utf-8" />
        <title>Exemplo</title>
        <link rel="stylesheet" href="css/exemplo.css">
    </head>
    <body>
        <div id="container">
            <div id="esquerda"> </div>
            <div id="direita"> </div>
            <div id="rodape"> </div>
        </div>
    </body>
</html>
```

Código HTML

```css
*{
    margin: 0px;
    padding: 0px;
}

body{
    width: 60vw;
    margin-left: 20vw;
    margin-top: 10vh;
}

#container{
    display: grid;

    grid-template-columns: 1fr 1fr;
    grid-template-rows: 60vh 25vh;
    grid-template-areas: "E D"
                         "R R";
    grid-gap: 10px;
}

#esquerda{
    grid-area: E;
    background-color: red;
}

#direita{
    grid-area: D;
    background-color: green;
}

#rodape{
    grid-area: R;
    background-color: blue;
}
```

Código CSS

- As Figuras 03 e 04 ilustram o comportamento da responsividade criada pelo código anterior.

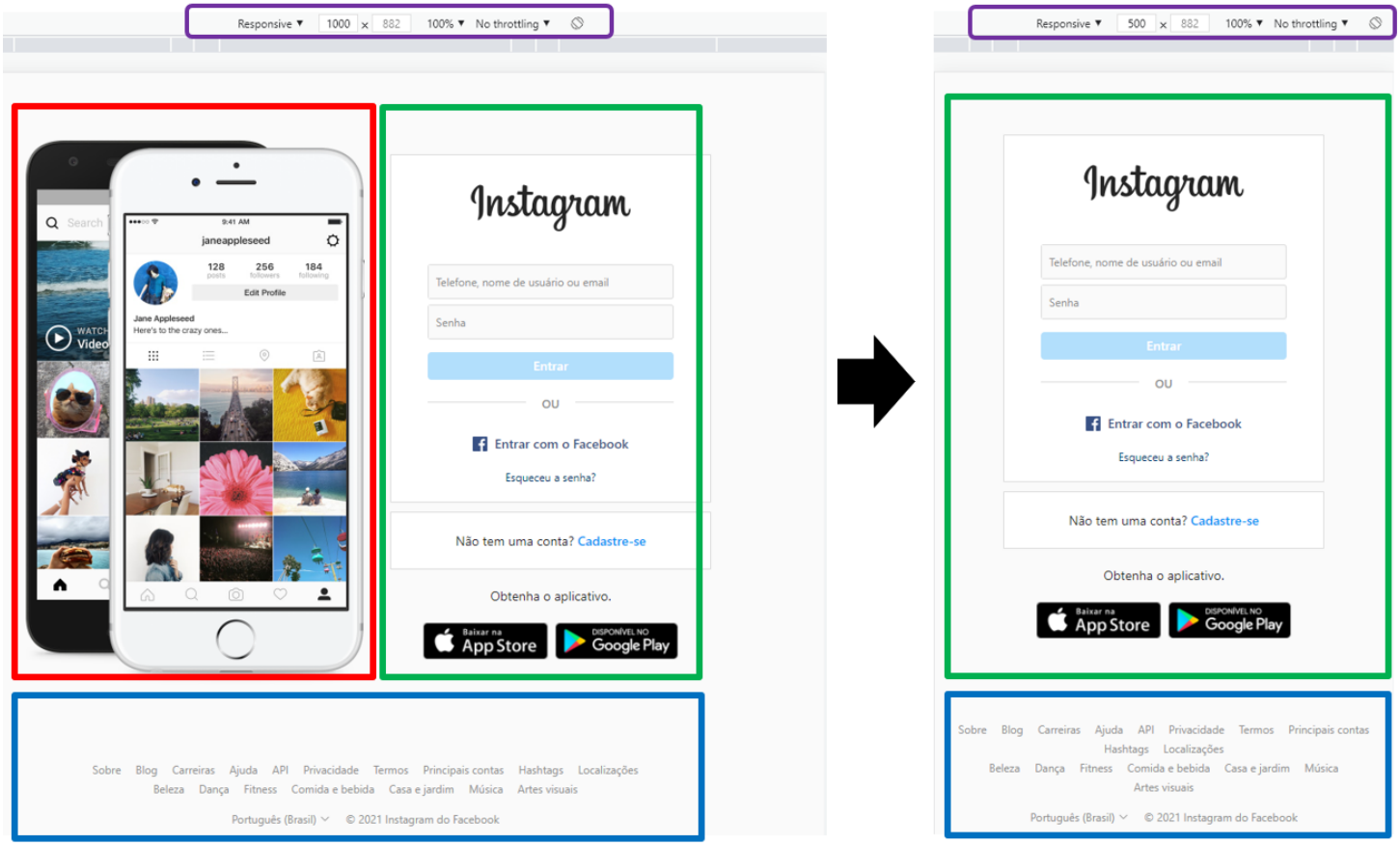

Figura 03: Responsividade na página do Instagram

Figura 04: Responsividade do exemplo

- O código abaixo ilustra a implementação completa do exemplo.

```
html>
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Exemplo</title>
        <link rel="stylesheet" href="css/exemplo.css">
    </head>
    <body>
        <div id="container">
            <div id="esquerda"> </div>
            <div id="direita"> </div>
            <div id="rodape"> </div>
        </div>
    </body>
</html>
```

O elemento ***‹meta name="viewport"...›*** permite ao programador acessar as informações sobre o ***viewport***. O ***viewport*** é a área onde o site será visualizado, e varia de acordo com o dispositivo do usuário. No código acima é definido que a largura do ***viewport*** é igual à largura do dispositivo e que o layout será exibido na largura inicial sem zoom.

Código HTML

```
*{
    margin: 0px;
    padding: 0px;
}

body{
    width: 60vw;
    margin-left: 20vw;
    margin-top: 10vh;
}

#container{
    display: grid;

    grid-template-columns: 1fr 1fr;
    grid-template-rows: 60vh 25vh;
    grid-template-areas: "E D"
                         "R R";
    grid-gap: 10px;
}

#esquerda{
    grid-area: E;
    background-color: red;
}

#direita{
    grid-area: D;
    background-color: green;
}

#rodape{
    grid-area: R;
    background-color: blue;
}

/* a media query será aplicada para ajustar o layout quando a tela for menor que 500px*/
@media screen and (max-width: 500px) {
    #container{
        grid-template-columns: 1fr;
        grid-template-rows: 60vh 25vh;

        grid-template-areas: "D"
                             "R";
    }

    #esquerda{
        display:none;
    }
}
```

O código ao lado segue a abordagem "**desktop-first**", na qual os estilos são inicialmente aplicados para desktop. As **media queries** utilizam a propriedade **max-width** para adaptar os estilos a telas menores.

Código CSS

Exercícios

1. Modifique o código do exemplo "**Layout do Instagram**", aplicando as alterações necessárias para que o layout responsivo siga a abordagem "**mobile-first**".

Figura 05: Protótipo do exercício 01

Dicas:

- **max-width** (abordagem Desktop-First): Define estilos que se aplicam **até** um determinado tamanho de tela (inclusive).
- **min-width** (abordagem Mobile-First): Define estilos que se aplicam **a partir** de um determinado tamanho de tela.
- Mobile-first – Porque utilizar e como fazer https://www.youtube.com/watch?v=SVwJA9aF2Cs
- Como escrever seu CSS para projetos mobile-first https://www.todoespacoonline.com/w/2015/03/como-escrever-seu-css-para-projetos-mobile-first/

2. Analise e abstraia o layout da página do Facebook. Em seguida, utilize as propriedades do Grid Layout e as Media Queries necessárias para criar uma página HTML responsiva que reproduza o layout ilustrado abaixo.

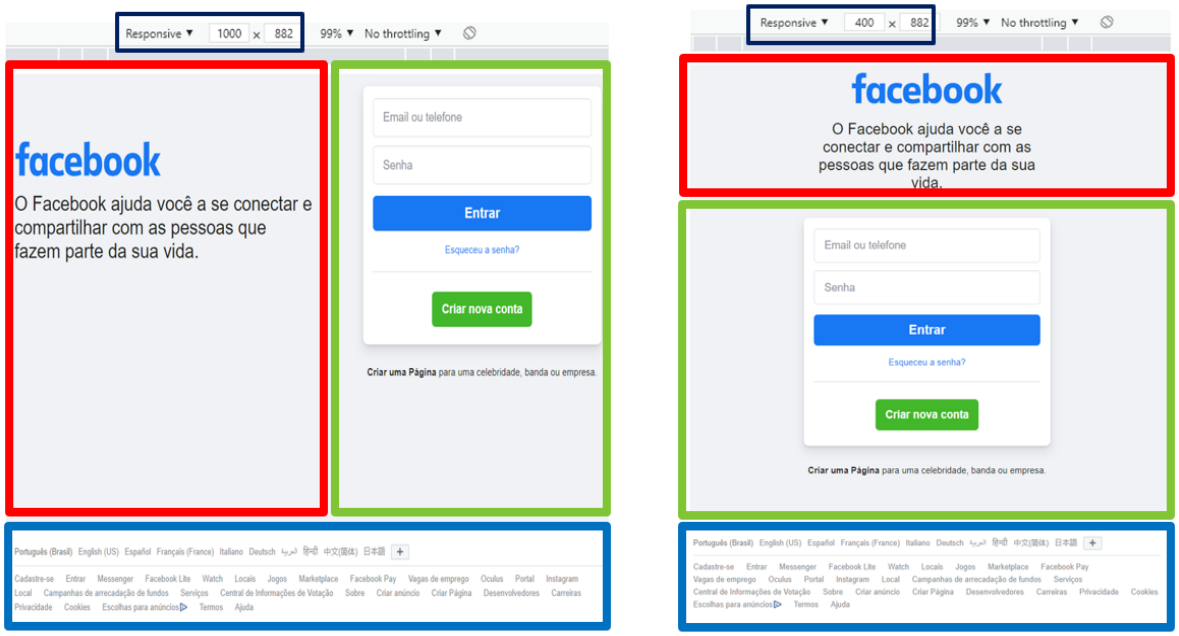

Figura 06: Protótipo do exercício 02

3. Utilize as propriedades do **Grid Layout** e as **Media Queries** necessárias para criar uma página HTML responsiva que reproduza o layout ilustrado abaixo.

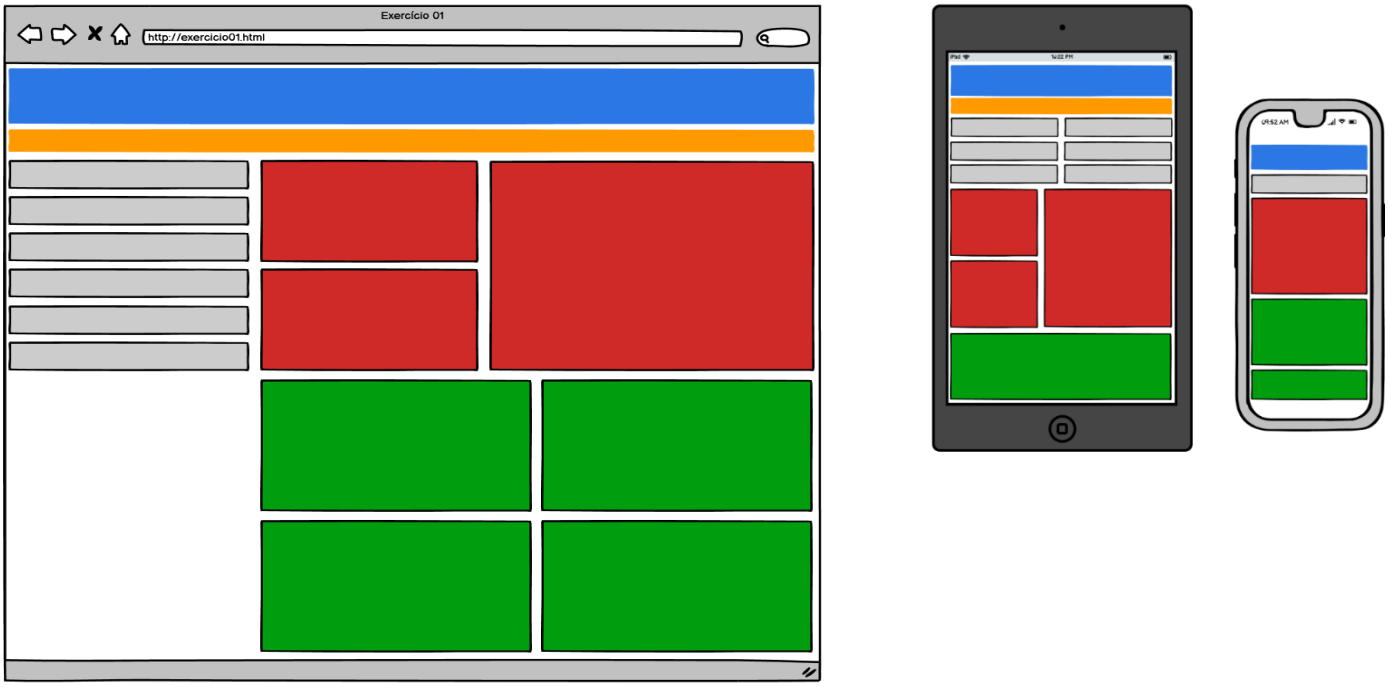

Figura 07: Protótipos do exercício 03

Observações:

- As quatro áreas de cor **verde** devem estar presentes em **todos** os layouts.
- Apenas a maior área de cor **vermelha** deve aparecer no layout para **celular**.
- Somente a primeira área de cor **cinza** deve ser exibida no layout para **celular**.
- Sugestão de larguras para cada dispositivo:
- Sugestões de Larguras (Breakpoints) para Media Queries

Dispositivo	Largura Sugerida
Celular	Até 576px (max-width: 576px)
Tablet	De 577px a 1024px (min-width: 577px e max-width: 1024px)
Computador	A partir de 1025px (min-width: 1025px)

4. Utilize o layout criado no exercício anterior e faça uma página semelhante à ilustração abaixo:

Observação:

O aluno tem total liberdade para escolher as imagens e notícias que considerar mais adequadas para a página. No entanto, caso prefira, pode utilizar as imagens e textos disponíveis no arquivo do guia

Figura 08: Protótipos do exercício 04

Material complementar

- O básico sobre Media Queries
https://desenvolvimentoparaweb.com/css/media-queries/

- Media queries, o que são e como usar no CSS?
https://www.treinaweb.com.br/blog/media-queries-o-que-sao-e-como-usar-no-css

- Media Queries e Breakpoints: o Básico e Fundamental
https://youtu.be/gYak6y7rbRw

- Site responsivo usando GRID Layout CSS!
https://www.youtube.com/watch?v=yLTFRELCh2o