```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="stylesheet" href="sass/css/extend.css" />
<title>Regra @extend</title>
</head>
<body>

<!--

            Regras de Estilo @extend

        O que faz a Regra de Estilo @extend?

    A regra de estilo @extend nos permite gerar um código inteligente onde podemos reaproveitar os estilos de uma
    classe pai adicionando estes estilos ás classes filhas desse elemento pai. Esse método é o que nós conhecemos
    como método BEM, uma metodologia que encoraja a criação de classes, com “sub-classes” que herdam as características
    da classe pai e ainda implementam uma classe modificadora específica para áquele elemento.

    A regra @extend veio para tornar essa metodologia fácil de ser implementada e utilizada, onde antes teríamos que
    criar classes e sub-classes na unha, ou utilizando extensões como o Next.js, que geram essas classes e sub-classes
    automaticamente, esbarrávamos em alguns problemas, como por exemplo a criação errada de sub-classes, sub-classes
    que não herdaram elementos das classes pai e a grande perca de tempo criando e verificando se todas as classes e
    sub-classes estão realmente de acordo.

    Algo importante que deve ficar marcado na nossa mente é que o @extend olha para os “seletores”, ele não foca em
    mixins ou functions, mas em seletores e faz a sua execução sempre em foco aos seletores e seus subseletores.


        Falando um pouco sobre a metodologia BEM:

    Essa metodologia é aplicada na nomenclatura das classes CSS dos nossos elementos HTML.
```

```
    A sigla BEM significa Block Element Modifier (Bloco, Elemento, Modificador), ela se vale de 3 pilares que formam a
    base dessa metodologia e também são categorias que utilizamos para dividir os nossos elementos de bloco HTML.

    Onde temos um  classe container "B" (Block), os elementos do container "E" (Elements) e os modificadores que serão
    comportamentos adicionais aos elementos dentro do bloco, como os pseudo-seletores: "hover", "active", "visited"
    entre outros.

    OBS: Para mais detalhes veja "Fundamentos do Sass nº15";

-->

    <div class="error">Houve um problema!</div>
    <div class="error--serious">Houve um problema GRAVE!</div> <!--Perceba que usamos uma nomenclatura BEM para a
                                                                    segunda div-->

</body>
</html>
```

ARQUIVO SASS...

```
.error {
    background-color: rgb(238, 95, 95);
    border: 1px solid red;
    margin: 10px;
    padding: 10px;
    &--serious {  /*Perceba aqui que usando o "&" o sistema já detectou que deve copiar as propriedades da classe
                    error para a classe error--serious*/
        @extend .error; /*Usamos o @extend para chamar a tensão da regra para o seletor "error"*/
        border-width: 3px;
    }
}
```

ARQUIVO CSS FINAL...

```css
@charset "UTF-8";
.error, .error--serious { /*Perceba no código css original que as propriedades foram copiadas em ambas as classes*/
  background-color: #ee5f5f;
  border: 1px solid red;
  margin: 10px;
  padding: 10px;
}
.error--serious { /*E somente a formatação adicional para a classe "error--serious" foi copiada para ela*/
  border-width: 3px;
}
```

RESULTADO NO NAVEGADOR...