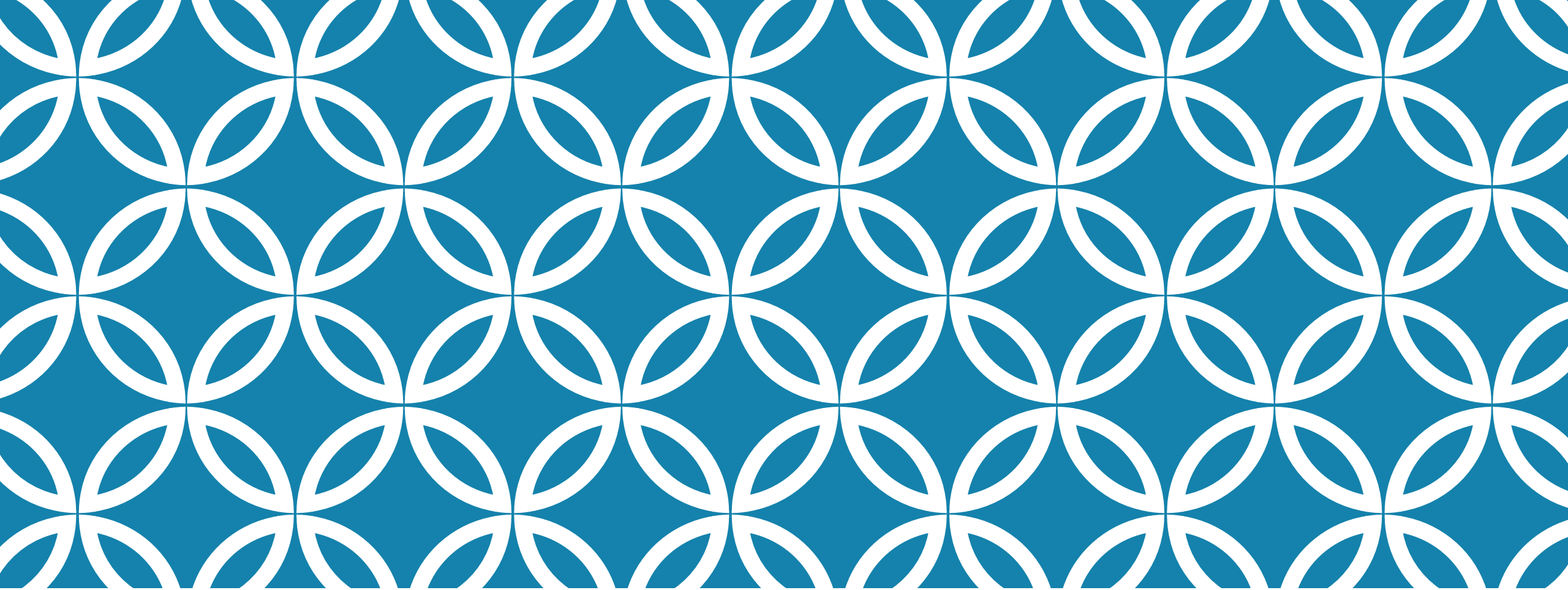
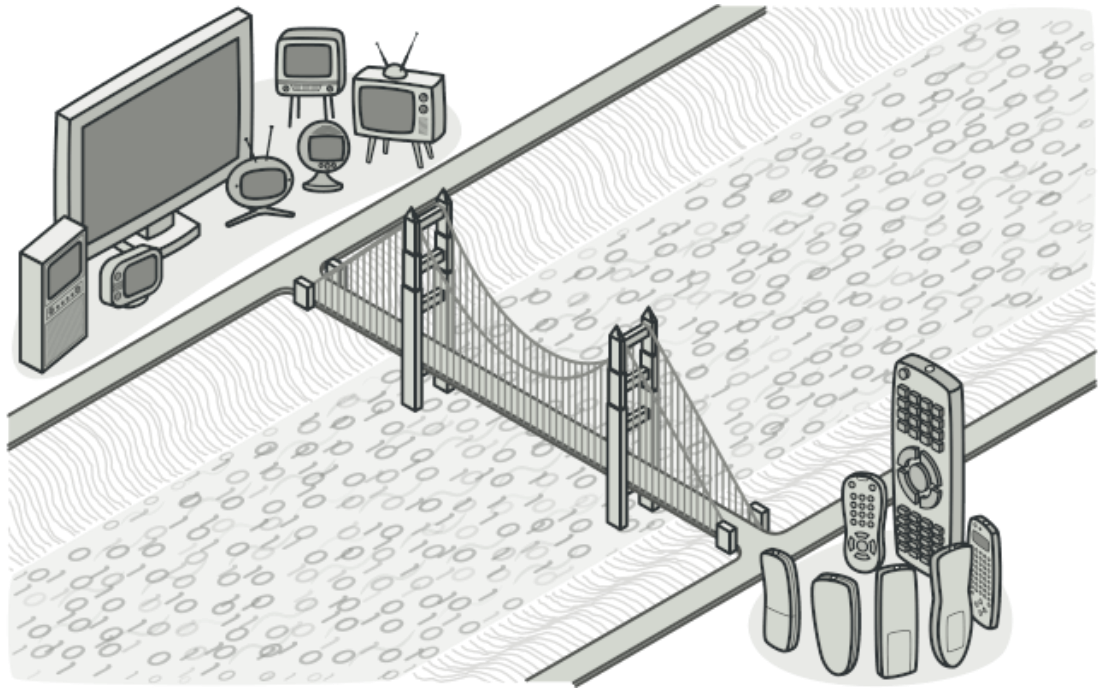


BRIDGE - PADRÃO DE PROJETO ESTRUTURAL

Gonçalo Garrido – CD470214

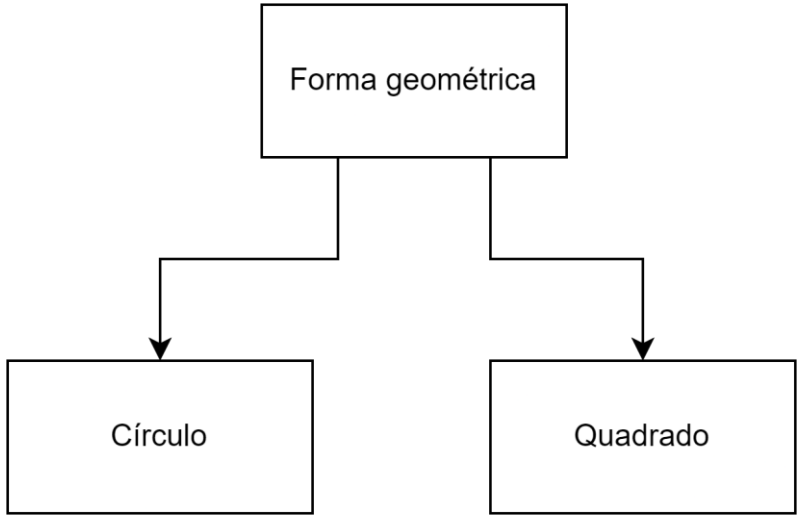
INTRODUÇÃO

O Bridge é um padrão de projeto estrutural que permite que você **divida uma classe grande** ou um **conjunto de classes intimamente relacionadas em duas hierarquias separadas** - abstração e implementação - que podem ser desenvolvidas independentemente uma da outra.



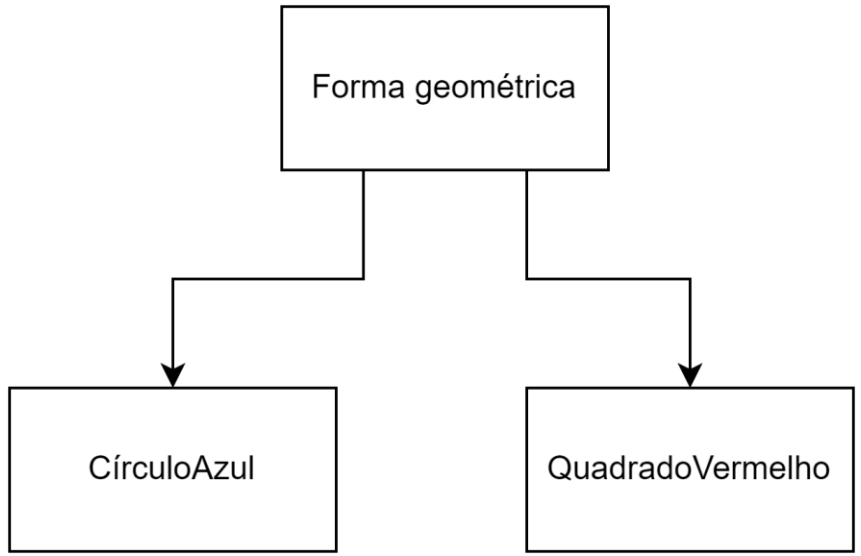
PROBLEMA

Imagina que temos uma classe Forma geométrica com subclasses, como **Círculo** e **Quadrado**. Agora, é necessário adicionar cores às formas, criando subclasses como **Vermelho e Azul**.



PROBLEMA

O problema surge quando se precisa criar combinações de subclasses para cada forma e cor, como **CírculoAzul** e **QuadradoVermelho**. À medida que novas formas e cores são adicionadas, o número de combinações cresce exponencialmente, tornando o código complexo e difícil de manter.

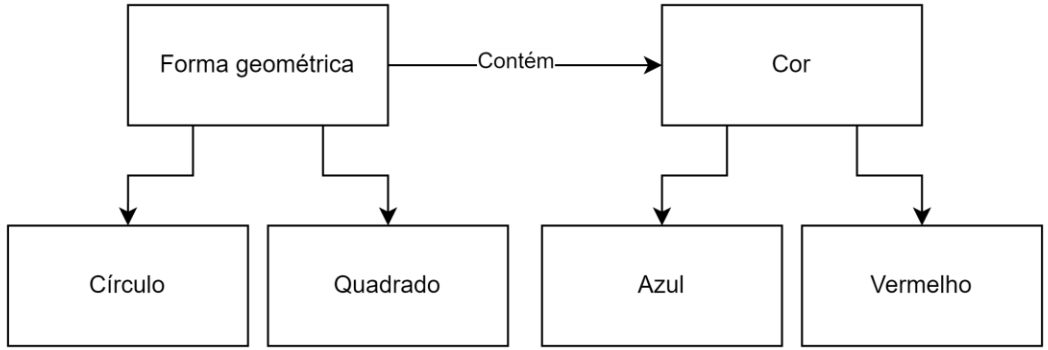


SOLUÇÃO

O padrão Bridge resolve esse problema substituindo a herança por composição.

Ele propõe extrair uma dimensão, como a cor, em uma hierarquia separada de classes, enquanto a classe original referencia um objeto dessa nova hierarquia.

Com isso, podemos adicionar novas formas e cores sem modificar a hierarquia existente, tornando o código mais flexível e modular.



CÓDIGO

```
using System;

// Implementação das Cores
public interface ICor {
  void Preencher();
}

// Implementações específicas de Cores
public class Vermelho: ICor {
  public void Preencher() {
    Console.WriteLine("Preenchendo com vermelho");
  }
}

public class Azul: ICor {
  public void Preencher() {
    Console.WriteLine("Preenchendo com azul");
  }
}

// Abstração das Formas Geométricas
public abstract class FormaGeometrica {
  protected ICor cor;

  public FormaGeometrica(ICor cor) {
    this.cor = cor;
  }

  public abstract void Desenhar();
}
```

```
// Implementações específicas de Formas Geométricas
public class Circulo: FormaGeometrica {
  public Circulo(ICor cor): base(cor) {}

  public override void Desenhar() {
    Console.WriteLine("Desenhando um círculo");
    cor.Preencher();
  }
}

public class Quadrado: FormaGeometrica {
  public Quadrado(ICor cor): base(cor) {}

  public override void Desenhar() {
    Console.WriteLine("Desenhando um quadrado");
    cor.Preencher();
  }
}

// Exemplo de uso
public class Program {
  public static void Main(string[] args) {
    ICor vermelho = new Vermelho();
    ICor azul = new Azul();

    FormaGeometrica circulo = new Circulo(vermelho);
    circulo.Desenhar();

    FormaGeometrica quadrado = new Quadrado(azul);
    quadrado.Desenhar();
  }
}
```

CONCLUSÃO

Em resumo, o padrão Bridge é uma solução elegante para lidar com o crescimento exponencial de combinações de classes ao adicionar novas funcionalidades. Ao separar a abstração da implementação, ele oferece flexibilidade, modularidade e facilita a manutenção do código.

OBRIGADO!!