Herança

Separar as partes que podem mudar das partes que não mudam. Exemplo: bibliotecas

- *Programador cliente deve poder usar o código sem a preocupação de ter que reescrever seu código caso surjam versões futuras*
- *Programador de biblioteca deve ter a liberdade de fazer melhoramentos sabendo que o cliente não terá que modificar o seu código*

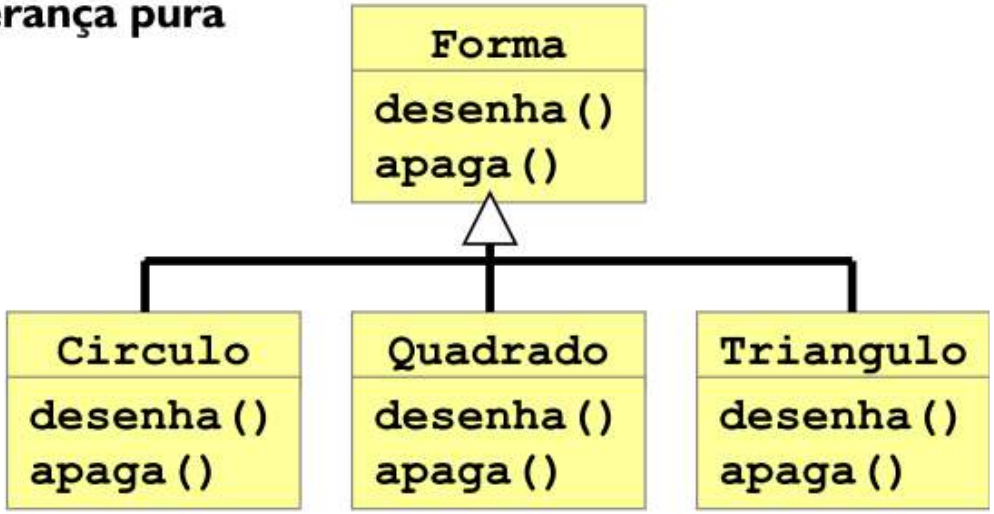
Em Java: esconder do cliente

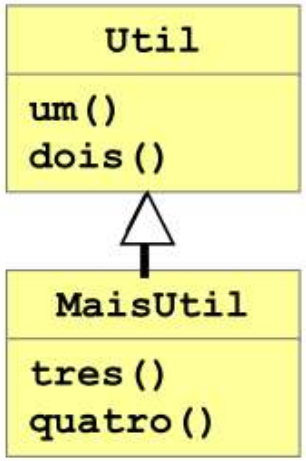
- *Métodos que não fazem parte da interface de uso*
- *Métodos que não fazem parte da interface de herança*
- *Todos os atributos de dados*

Quando você precisa de uma classe, você pode

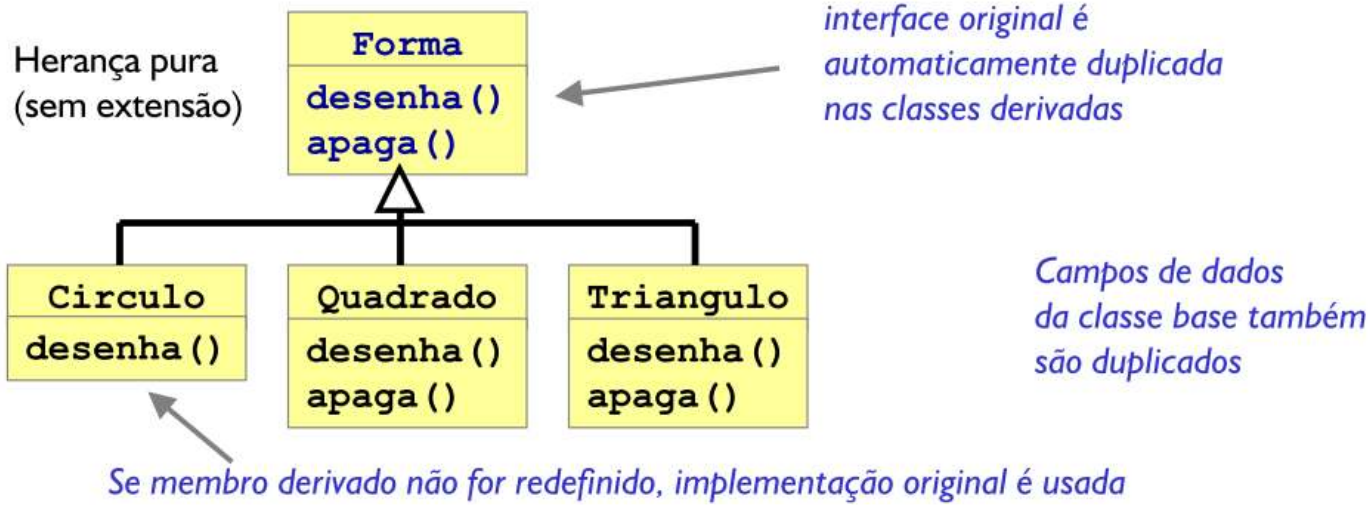
- *Usar uma classe que faz exatamente o que você deseja fazer*
- *Escrever uma classe do zero*
- *Reutilizar uma classe existente com composição*
- *Reutilizar uma classe existente ou estrutura de classes com herança*

Herança pura





Extensão




```
class Forma {
  public void desenha() {
    /*...*/
  }
  public void apaga() {
    /*...*/
  }
}
```

```
class Circulo extends Forma {
  public void desenha() {
    /* nova implementação */
  }
}
```

Assinatura do método tem que ser igual ou sobreposição não ocorrerá (poderá ocorrer sobrecarga não desejada)