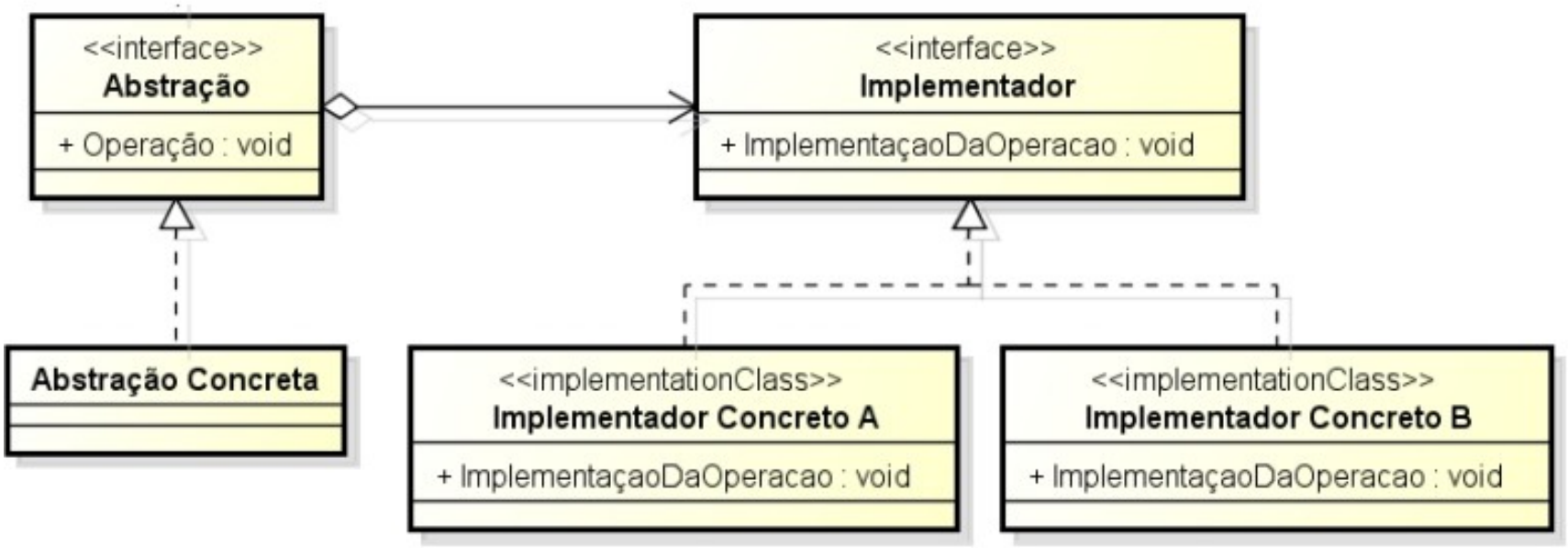
Bridge

Instituto Federal do Espirito Santo – Campus Colatina
Disciplina: Projeto de Sistemas
Professor: Giovany Frossard Teixeira

Elementos Essenciais

- **Nome:** *Bridge (Handle/Body)*
- **Problema:** Uma abstração pode ter várias implementações possíveis, mas a herança (maneira usualmente utilizada) liga uma implementação à abstração de forma permanente o que pode gerar dificuldade de aumentar, reutilizar ou modificar o código.
- **Solução:** Desacoplar uma abstração da sua implementação, de modo que as duas possam variar independentemente.
- **Consequências:** Maior facilidade de modificar, reutilizar ou acrescentar novas funcionalidades ao sistema. Maior extensibilidade decorrente do desacoplamento desse padrão.

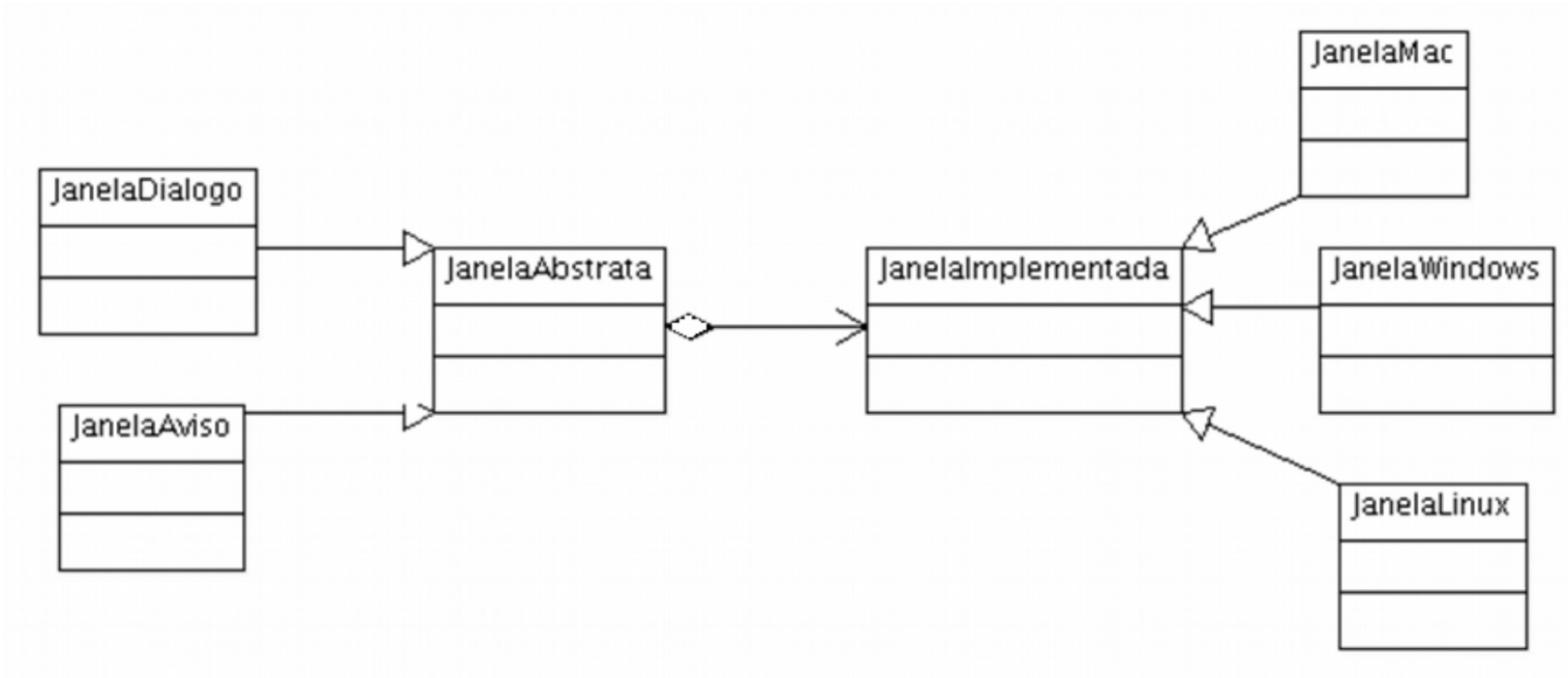
Padrão Bridge



Fonte: https://brizeno.wordpress.com/category/padroes-de-projeto/bridge/
Acessado em 09/03/2020

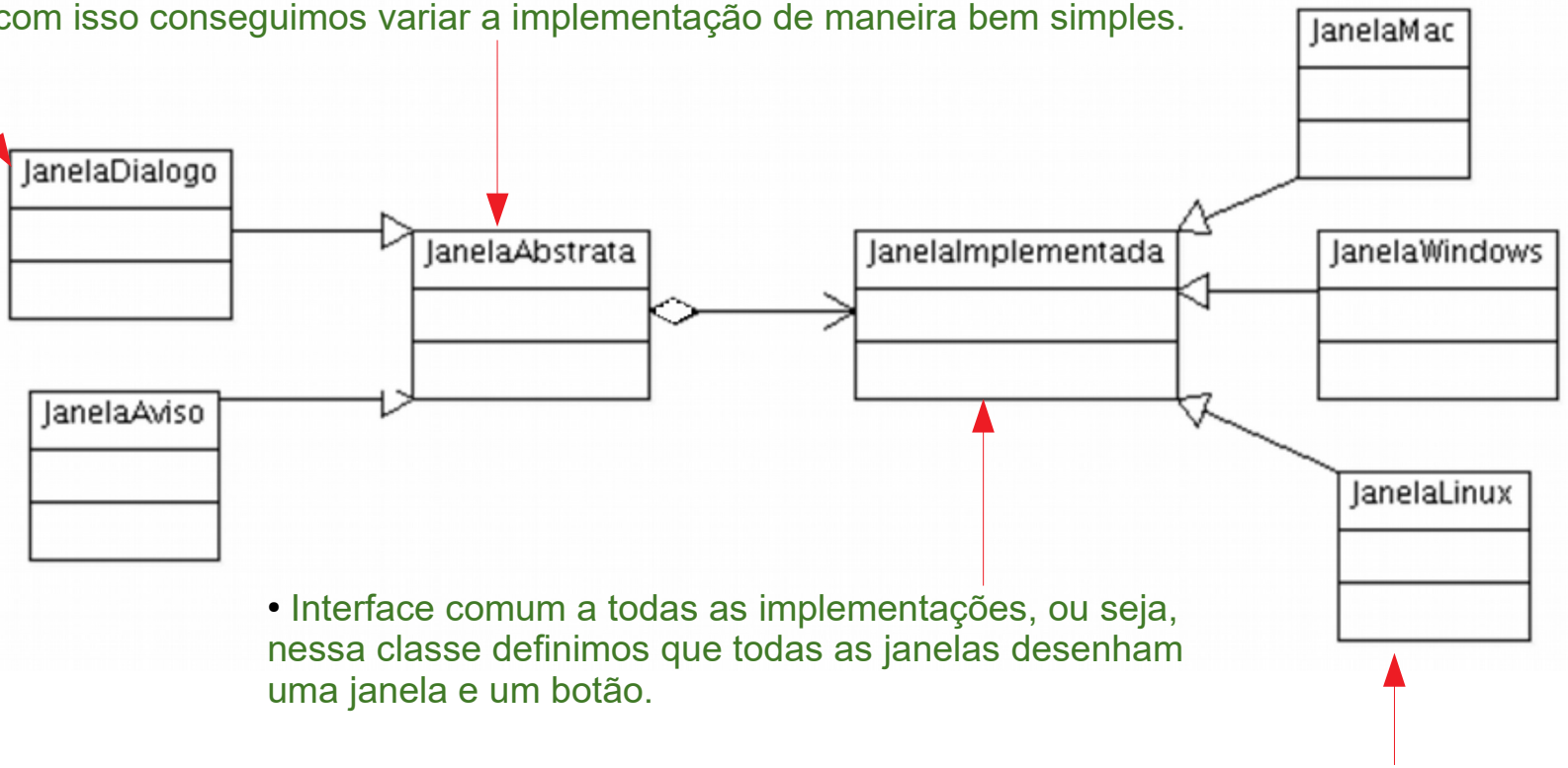
Problema

- Suponha que é necessário fazer um programa que vá funcionar em várias plataformas, por exemplo, Windows, Linux, Mac, etc. O programa fará uso de diversas abstrações de janelas gráficas, por exemplo, janela de diálogo, janela de aviso, janela de erro, etc.



• Uma janela de diálogo exibe sempre três botões: Sim, Não e Cancelar.
Ou seja, independente de qual plataforma está sendo utilizada,
a abstração é sempre a mesma.

• Fornecer uma interface de acesso comum para as abstrações de janelas.
• Essa classe possui uma referência para a interface das janelas implementadas,
com isso conseguimos variar a implementação de maneira bem simples.



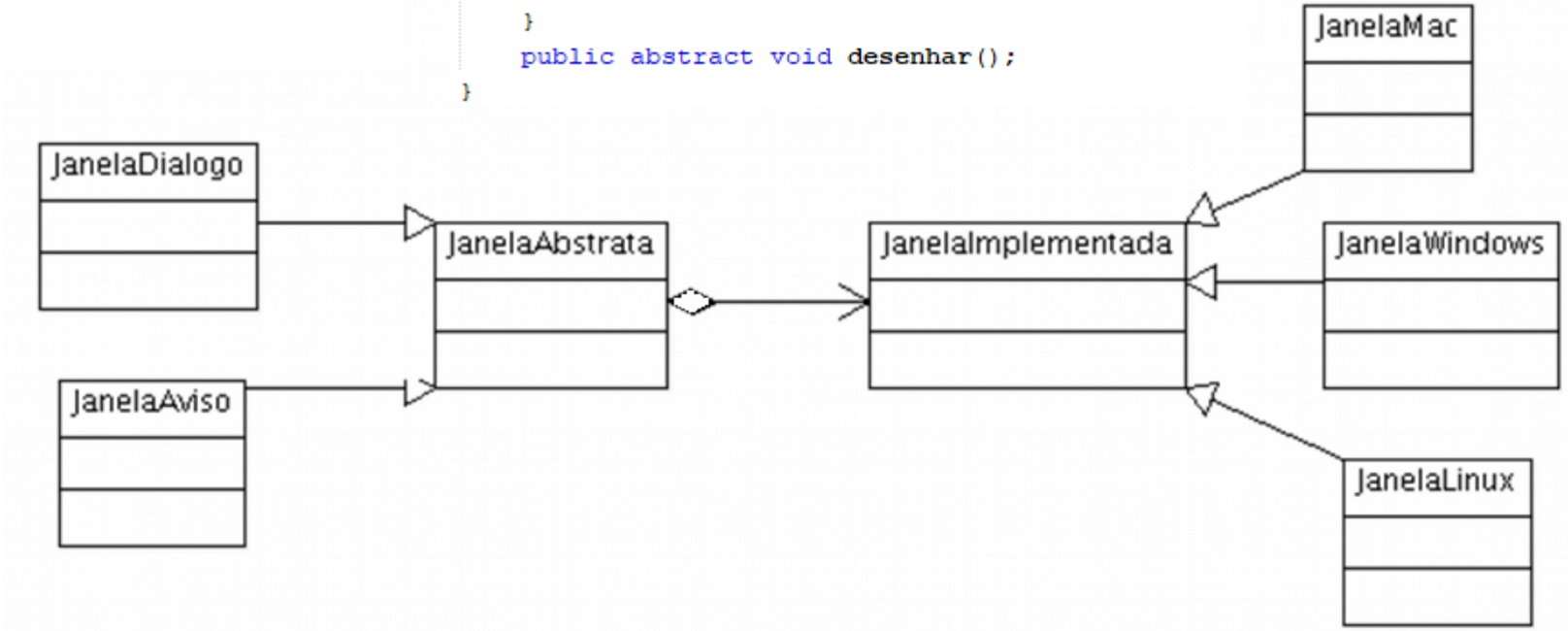
• Interface comum a todas as implementações, ou seja,
nessa classe definimos que todas as janelas desenham
uma janela e um botão.

• Classe concreta que desenha a janela
na plataforma Linux.

Abstrações

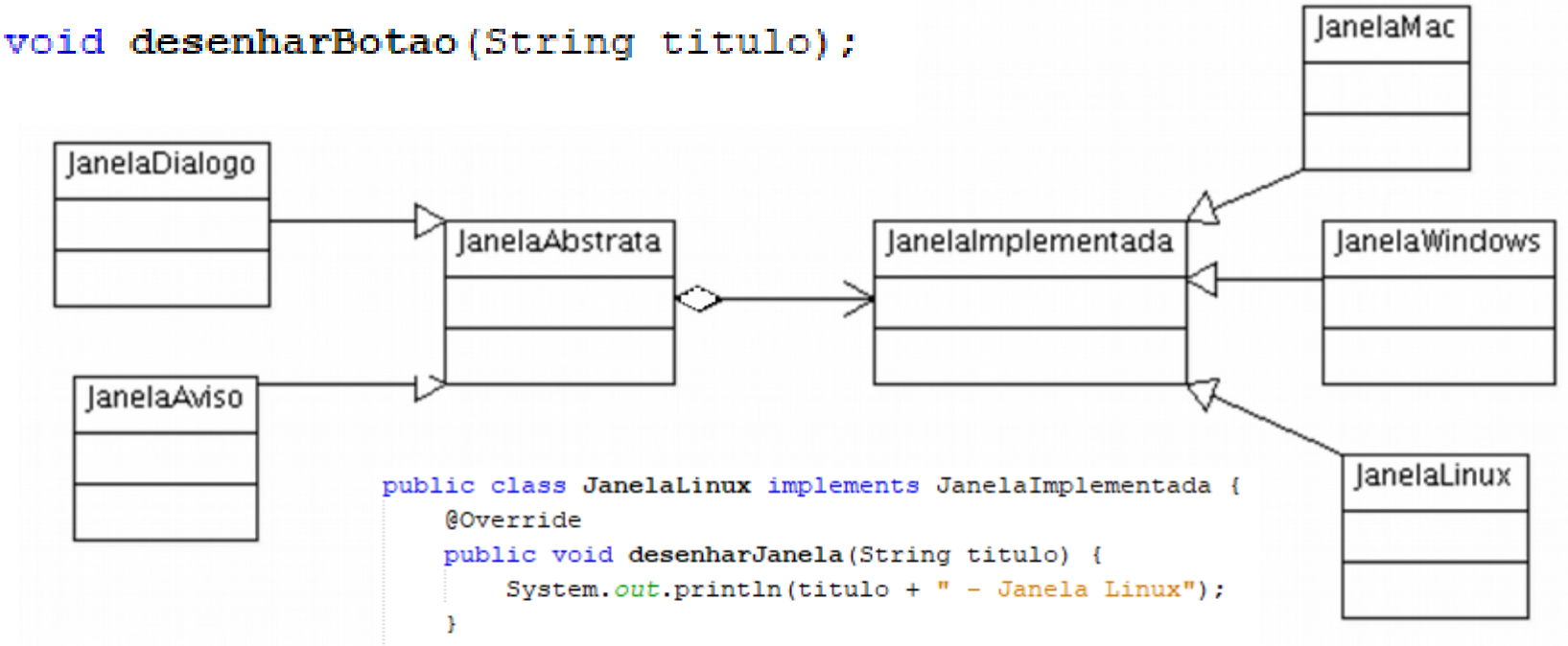
```java
public abstract class JanelaAbstrata {

    protected JanelaImplementada janela;

    public JanelaAbstrata(JanelaImplementada j) {
        janela = j;
    }

    public void desenharJanela(String titulo) {
        janela.desenharJanela(titulo);
    }

    public void desenharBotao(String titulo) {
        janela.desenharBotao(titulo);
    }
    public abstract void desenhar();
}
```

```java
public class JanelaAviso extends JanelaAbstrata {
    public JanelaAviso(JanelaImplementada j) {
        super(j);
    }

    @Override
    public void desenhar() {
        desenharJanela("\n\nJanela de Aviso");
        desenharBotao("Ok");
    }
}
```



Implementações

```
public interface JanelaImplementada {
    void desenharJanela(String titulo);

    void desenharBotao(String titulo);
}
```




```
public class JanelaLinux implements JanelaImplementada {
    @Override
    public void desenharJanela(String titulo) {
        System.out.println(titulo + " - Janela Linux");
    }

    @Override
    public void desenharBotao(String titulo) {
        System.out.println(titulo + " - Botão Linux");
    }
}
```

main

```
public class Bridge {

    public static void main(String[] args) {
        JanelaAbstrata janela = new JanelaDialogo(new JanelaLinux());
        janela.desenhar();

        janela = new JanelaAviso(new JanelaLinux());
        janela.desenhar();

        janela = new JanelaDialogo(new JanelaWindows());
        janela.desenhar();
    }
}
```

```
Janela de Diálogo - Janela Linux
Botão Sim - Botão Linux
Botão Não - Botão Linux
Botão Cancelar - Botão Linux


Janela de Aviso - Janela Linux
Ok - Botão Linux


Janela de Diálogo - Janela Windows
Botão Sim - Botão Windows
Botão Não - Botão Windows
Botão Cancelar - Botão Windows
BUILD SUCCESSFUL (total time: 1 second)
```

Considerações finais

- O padrão Adapter é utilizado para fazer classes não relacionadas trabalharem em conjunto, ele normalmente é utilizado em sistemas já projetados anteriormente. Já o padrão Bridge pode ser utilizado em um projeto, desde o início, para fazer com que abstrações e implementações possam variar independentemente.
- O padrão Bridge pode ser encontrado em implementações de desenho gráfico (Java Swing, AWT, etc), em pontes de conexão com banco de dados – JDBC.
- No padrão Bridge uma implementação não fica permanentemente presa a uma interface, a implementação pode ser configurada em tempo de execução.
- É possível estender as hierarquias de **Abstração** e **Implementador** independentemente.

Dúvidas?