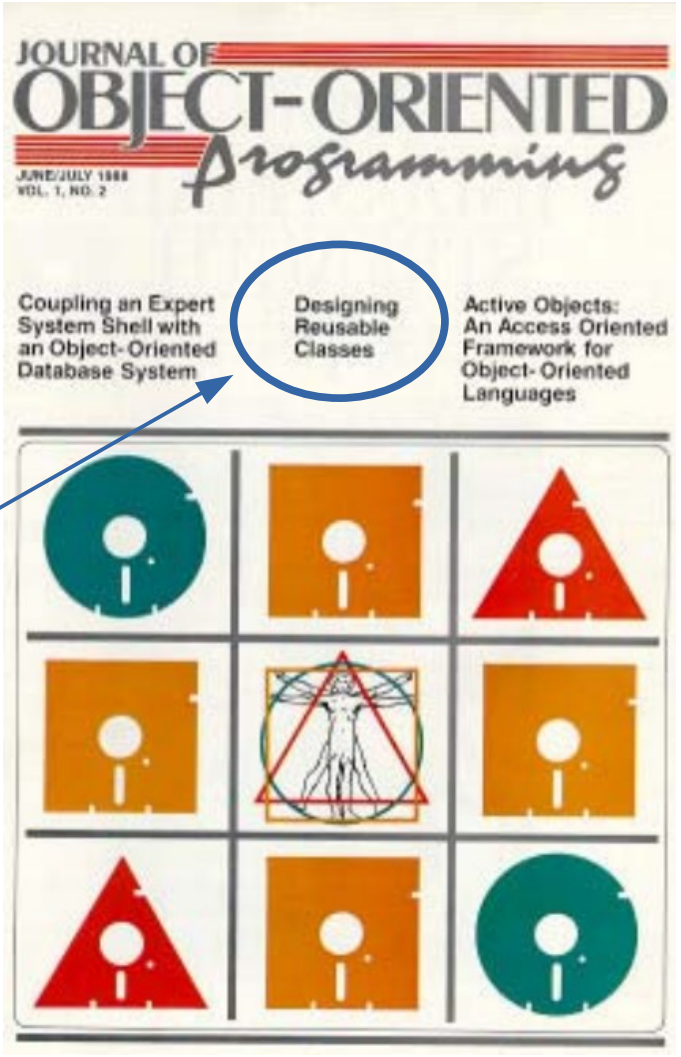
Reusabilidade de software

“Designing reusable classes”

IFC Bnu - Prof. Hylson



Material

JOHNSON, R.; FOOTE, B. **Designing Reusable Classes**. Journal of Object-oriented Programming, v. 1, n. 1, p. 22-35, 1988. **2149** citações em 21/04/2021.

2234 em 04/04/22

2236 em 11/04/22

2354 em 19/02/24

University of Illinois

Laputan Laboratories, Ltd.

*I've worked (and am working) with many OO languages, but **SmallTalk** is still my favorite.*

JOHNSON, 1988

1. Introdução
2. Programação orientada a objetos
3. Reuso de software

Seção 1: introdução

Introdução

Programação orientada a objetos é sempre elogiada por promover o reuso de software

Introdução

Programação orientada a objetos é sempre elogiada por promover o reuso de software

Reduz... 

tempo de desenvolvimento
custo de manutenção

Introdução

Programação orientada a objetos é sempre elogiada por promover o reuso de software

Reduz...

tempo de desenvolvimento
custo de manutenção

Simplifica...

criação de novos sistemas
novas versões de sistemas antigos

Introdução

Programação orientada a
ob
pr

Programação orientada a objetos não é uma PANACÉIA

Reduz... Simplifica...

Programador-líder de Doom, Quake, etc

John Carmack @ID_AA_Carmack · Mar 31, 2011

Sometimes, the elegant implementation is just a function. Not a method. Not a class. Not a framework. Just a function.

Introdução

Técnicas de projeto tornam o software OO mais reusável

Introdução

Técnicas de projeto tornam o software OO mais reusável

Introdução

Abstração de dados

Sistema modular

Mais fácil para compreender

Introdução

Abstração de dados

Sistema modular

Mais fácil para compreender

Herança

Classes compartilham métodos

Programação-por-diferença

Introdução

Framework

Um conjunto de classes que incorpora um projeto abstrato para soluções de uma família de problemas relacionados...

Introdução

Framework

Um conjunto de classes que incorpora um projeto abstrato para soluções de uma família de problemas relacionados...

…e provê reuso em uma granularidade maior do que classes.

Introdução

Framework

Durante fases iniciais do framework, o engenheiro de software precisa conhecer como o componente funciona para reutilizá-lo...

Introdução

Framework

Durante fases iniciais do framework, o engenheiro de software precisa conhecer como o componente funciona para reutilizá-lo...

…mas o framework refinado é uma “caixa preta”: componentes podem ser reusados sem conhecimento de sua implementação.

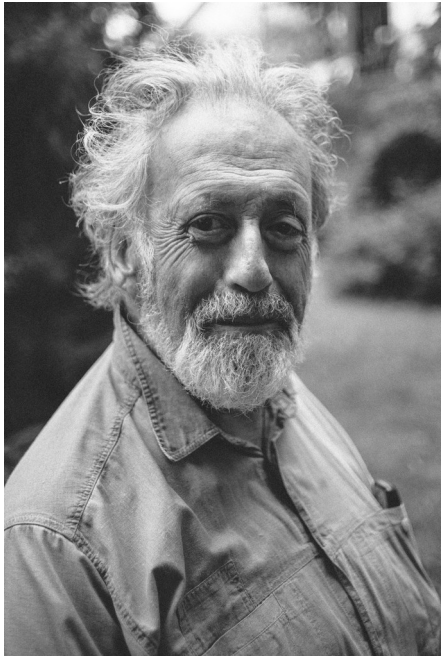
Introdução

Projeto de classes reusáveis requer...

Julgamento

Experiência

Experimentação



Introdução

Projeto de classes reusáveis requer...

Julgamento

Experiência

Experimentação

Os conceitos a seguir tem o objeto de ajudar novos projetistas a adquirir essas habilidades mais rapidamente

Seção 2: programação orientada a objetos

POO

Objeto encapsula dados e operações sobre esses dados

POO

Objeto encapsula dados e
operações sobre esses dados

Linguagem OO:
facilidades *para*
realizar abstração
de dados

Como as linguagens OO
se diferenciam das
linguagens convencionais?

POO

Como as linguagens OO
se diferenciam das
linguagens convencionais?

POO

Como as linguagens OO
se diferenciam das
linguagens convencionais?

Polimorfismo

*(ligação tardia
de chamada de
procedimento*

Herança

POO

Polimorfismo

Operações são executadas em objetos enviando-se mensagens a esses objetos

POO

Polimorfismo

Operações são executadas em objetos enviando-se mensagens a esses objetos

Um envio de mensagem é implementado buscando-se o método correto na classe destinatária e invocando esse método.

POO

Polimorfismo

O envio de mensagens causa o polimorfismo.

Um envio de mensagem é implementado buscando-se o método correto na classe destinatária e invocando esse método.

POO

Polimorfismo

O envio de mensagens causa o polimorfismo.

Exemplo: um método que soma elementos em um vetor funciona se todos os elementos do vetor *compreenderem* a mensagem de adição, independente da classe a que pertençam.

POO

Polimorfismo

Em Smalltalk, um objeto pode acessar um vetor no qual cada elemento é de uma classe diferente.

POO

Polimorfismo

Em Smalltalk, um objeto pode acessar um vetor no qual cada elemento é de uma classe diferente.

Como todos os elementos do vetor entendem o mesmo conjunto de mensagens, o objeto pode interagir com os elementos do vetor a despeito de suas classes.

POO

Polimorfismo

Em vez de…
If número: soma númerica
If texto: concatenação
If real: soma com arredondamento
...

POO

Polimorfismo

Em vez de…
If número: soma númerica
If texto: concatenação
If real: soma com arredondamento
...

“Some-se a este outro objeto”
O próprio objeto conhece como deve “ser somado”

POO

Protocolo

A especificação de um objeto é dada por seu protocolo.

Um conjunto de mensagens que podem ser enviadas ao objeto.

POO

Protocolo

Protocolos são importantes para programadores se comunicarem.

POO

Protocolo

Protocolos são importantes para programadores se comunicarem.

Protocolos compartilhados criam um vocabulário que facilita o reuso e a aprendizagem de novas classes.

POO

Protocolo

Protocolos são importantes para programadores se comunicarem.

Protocolos compartilhados criam um vocabulário que facilita o reuso e a aprendizagem de novas classes.

Programadores usam o mesmo nome para operações em classes diferentes.

POO

Protocolo

Linguagens procedurais precisam criar nomes diferentes para operações similares, para que essas operações possam ser usadas em um mesmo programa.

SomaMatriz, SomaComplexos, SomaPolinomio

POO

Herança

Cada classe possui uma superclasse da qual herda operações e estruturas internas.

POO

Herança

Cada classe possui uma superclasse da qual herda operações e estruturas internas.

Novas classes...

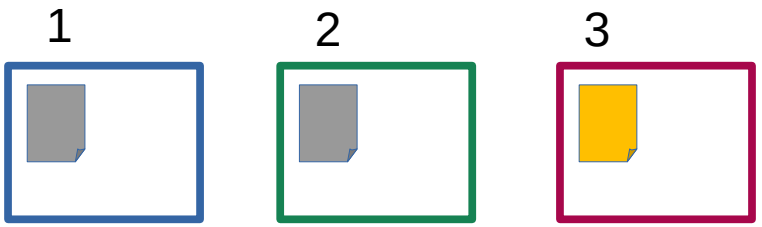
…podem adicionar operações
...podem redefinir operações
…**NÃO** podem excluir operações herdadas

POO

Herança

Vantagens...

Código similar em classes diferentes pode ser movido para uma superclasse.



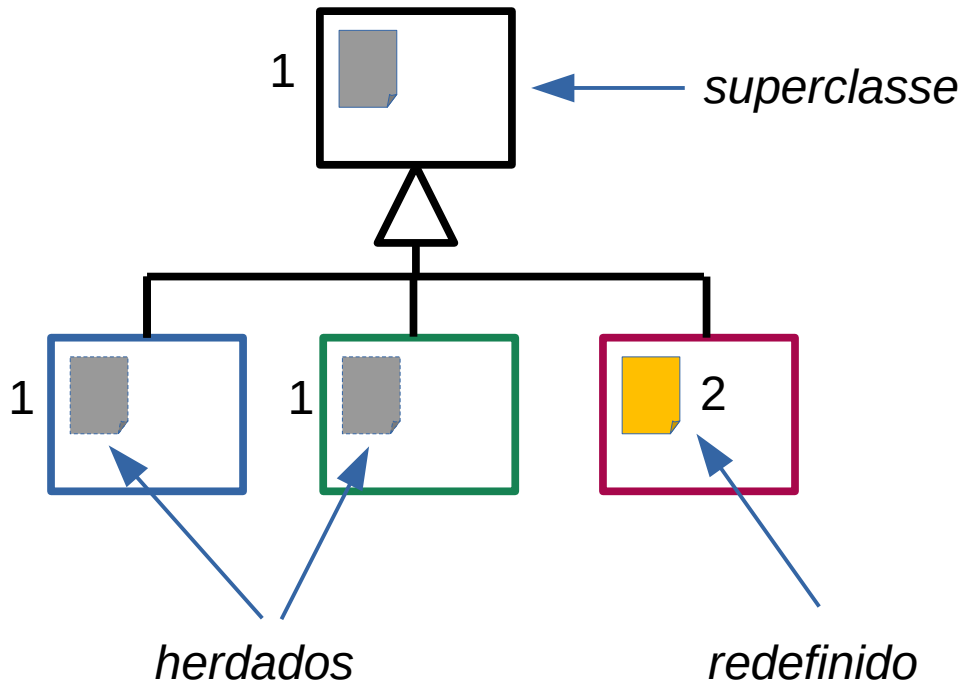
“**3 códigos**” (1 e 2 são iguais)

POO

Herança

Vantagens...

Código similar em classes diferentes pode ser movido para uma superclasse.



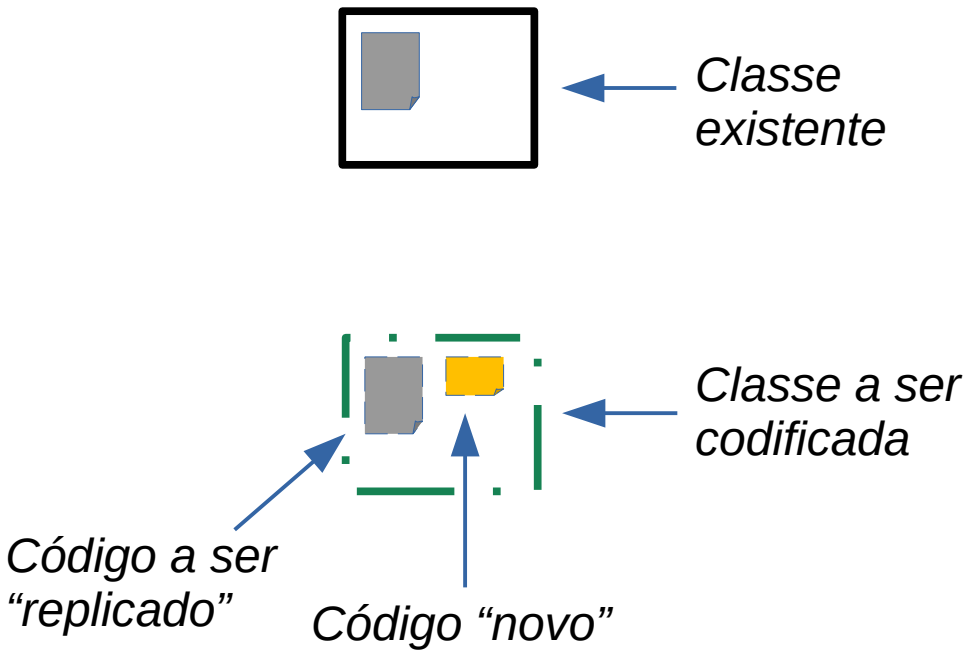
“**2 códigos**”

POO

Herança

Vantagens...

Programação-por-diferenças: novas classes criadas a partir de outras relacionadas.

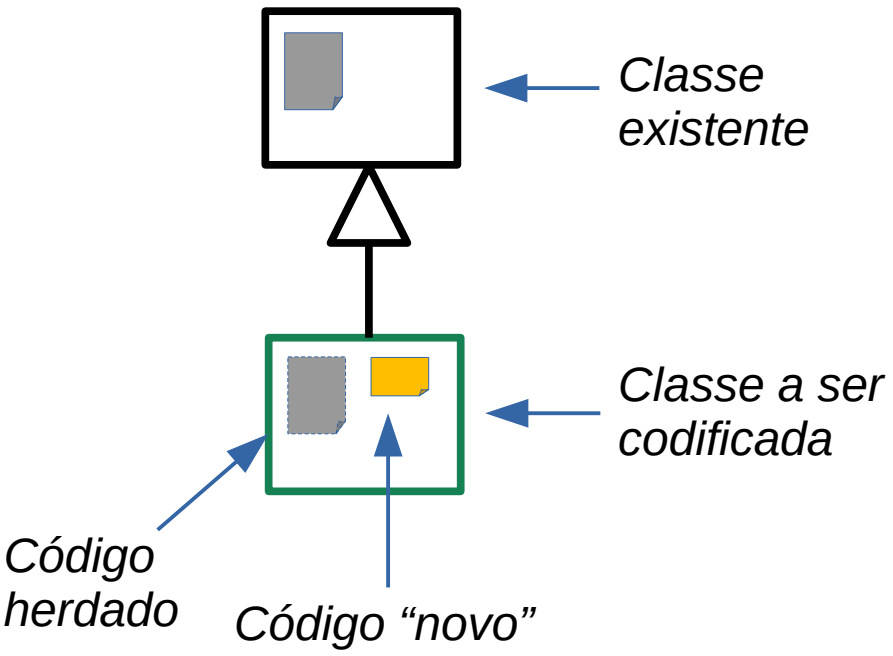


POO

Herança

Vantagens...

Programação-por-diferenças: novas classes criadas a partir de outras relacionadas.



POO

Herança

Vantagens...

Facilita a criação de protocolos padronizados.

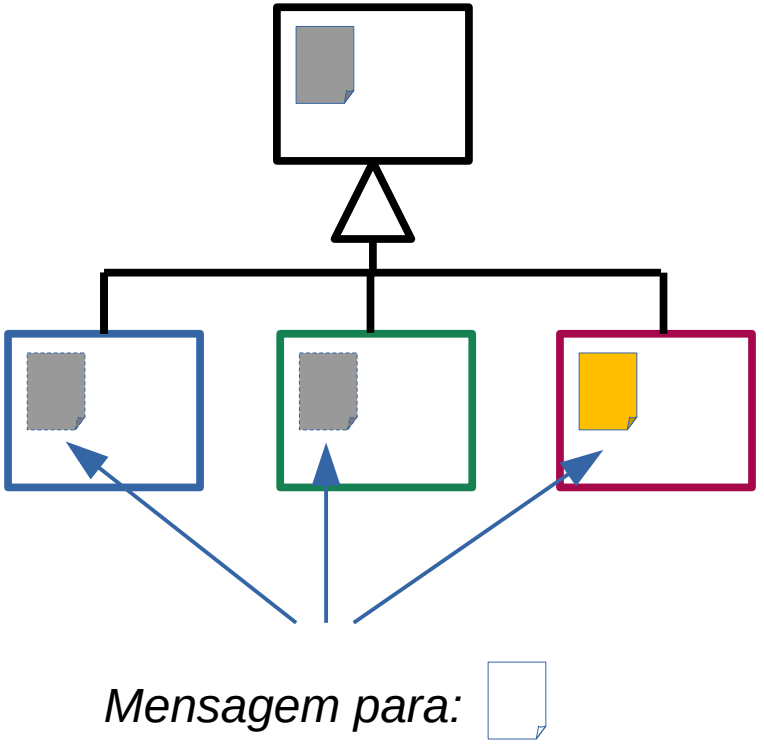
Uma mensagem pode ser enviada para objetos a que conhecem

POO

Herança

Vantagens...

Facilita a criação de protocolos padronizados.

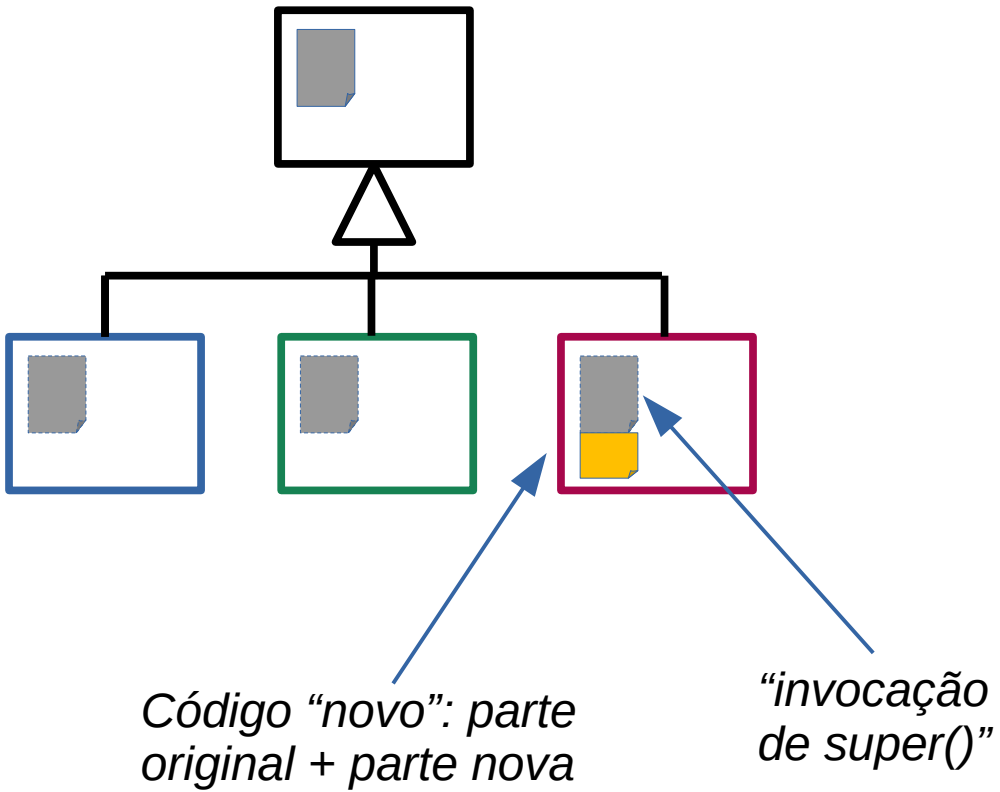


POO

Herança

Vantagens...

Extensões mantém o código original intacto.



POO

Classe abstrata

Não possui instâncias

Classes no topo de hierarquias geralmente são abstratas

Geralmente não define variáveis de instância

(atributos)

Define métodos a serem implementados em subclasses

POO

Classe abstrata

Exemplo: Coleção

Métodos: selecionar, inserir

Subclasses: Vetor, Conjunto, Dicionário

POO

Classe abstrata

Uma classe que não é abstrata é concreta.

POO

Classe abstrata

Uma classe que não é abstrata é concreta.

Herança a partir de classes concretas inclui representação de dados, o que pode conflitar com a subclasse

POO

Classe abstrata

Uma classe que não é abstrata é concreta.

Herança a partir de classes concretas inclui representação de dados, o que pode conflitar com a subclasse

Sempre que possível, herdar de classe abstrata

POO

Classe abstrata

Não é fácil criar classes abstratas

Hierarquia de classes concretas pode ser reorganizada para encontrar a classe abstrata “oculta”

POO

Classe abstrata

Exemplo:
classe View
controla uma
região da tela

* Contém 27 subclasses
* É concreta

POO

Classe abstrata

Exemplo:
classe View controla uma região da tela

* Contém 27 subclasses
* É concreta

Análise revela: algumas premissas de View não são usadas em todas as subclasses

POO

Classe abstrata

Exemplo: classe View controla uma região da tela

Views pode possuir subviews

Análise revela: algumas premissas de View não são usadas em todas as subclasses

POO

Classe abstrata

Exemplo: classe View controla uma região da tela

Views pode possuir subviews

Códigos em View lidam com adição e posicionamento de subviews

Análise revela: algumas premissas de View não são usadas em todas as subclasses

POO

Classe abstrata

Exemplo: classe View controla uma região da tela

Views pode possuir subviews

Códigos em View lidam com adição e posicionamento de subviews

View precisa posicionar subview?

O que é essencial para uma View? Informações a mais confudem novos usuários da classe.

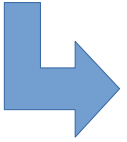
POO

Classe abstrata

Exemplo: classe View controla uma região da tela

Solução: dividir

Classe View, abstrata



Subclasse ViewWithSubview, concreta, com habilidade de manipular subviews

POO

Herança x Decomposição

Herança é um recurso poderoso...

POO

Herança x Decomposição

Herança é um recurso poderoso...

...mas bastante usado incorretamente

POO

Herança x Decomposição

Herança é um recurso poderoso...

...mas bastante usado incorretamente

Às vezes uma subclasse...

POO

Herança x Decomposição

Herança é um recurso poderoso...

...mas bastante usado incorretamente

Às vezes uma subclasse...

...seria melhor como componente.

POO

Herança x Decomposição

Exemplo:
Janela
gráfica

POO

Herança x Decomposição

Exemplo: Janela gráfica

Janela pode ser subclasse de retângulo...

POO

Herança x Decomposição

Exemplo: Janela gráfica

Janela pode ser subclasse de retângulo...

...mas retângulo pode ser uma variável de instância de uma janela.

Faz sentido mover uma janela, mas por que mover um retângulo?

POO

Herança x Decomposição

Comportamentos podem ser mais fáceis para reusar como um componente do que por herança.

Exemplo:
barra de
rolagem

POO

Herança x Decomposição

Comportamentos podem ser mais fáceis para reusar como um componente do que por herança.

Exemplo: barra de rolagem

Como uma característica herdada: difícil para converter uma classe com rolagem vertical em outra sem rolagem, ou com relagem horizontal

POO

Herança x Decomposição

Comportamentos podem ser mais fáceis para reusar como um componente do que por herança.

*Mais fácil fazer barras de rolagem como **componentes** do objeto que precisa ser “rolado”*

Exemplo: barra de rolagem

Como uma característica herdada: difícil para converter uma classe com rolagem vertical em outra sem rolagem, ou com relagem horizontal

POO

Herança x Decomposição

Sistemas possuem hierarquias
de **classes** e **instâncias**

Exemplo:
árvore de
janelas

POO

Herança x Decomposição

Sistemas possuem hierarquias de **classes** e **instâncias**

Exemplo: árvore de janelas

Cada subjanela é uma instância de janela (View)

POO

Herança x Decomposição

Sistemas possuem hierarquias de **classes** e **instâncias**

Exemplo: árvore de janelas

Cada subjanela é uma subclasse de janela (View)

A janela raiz é uma instância de StandardSystemView (Smalltalk-80)

Seção 3: reuso de software

Reuso de software

Desenvolver sistemas é caro.

Reuso de software

Desenvolver sistemas é caro.

Manter sistemas é ainda mais caro!

Reuso de software

Desenvolver sistemas é caro.

Manter sistemas é ainda mais caro!

Uma forma de reuso é melhorar sistemas

Reuso de software

Desenvolver sistemas é caro.

Manter sistemas é ainda mais caro!

Uma forma de reuso é melhorar sistemas

Manutenção é uma forma de reuso!

Reuso de software

Desenvolver sistemas é caro.

Manter sistemas é ainda mais caro!

Uma forma de reuso é melhorar sistemas

Manutenção é uma forma de reuso!

Melhorar ou manter sistemas requer programadores que compreendam e modifiquem software desenvolvido por outros.

Difícil!

Reuso de software

Manutenção de software...

Corretiva Diagnosticar e corrigir erros

Adaptativa

Perfectiva

Reuso de software

Manutenção de software...

Corretiva	Diagnosticar e corrigir erros
Adaptativa	Adaptar sistema a novo hardware
Perfectiva	

Reuso de software

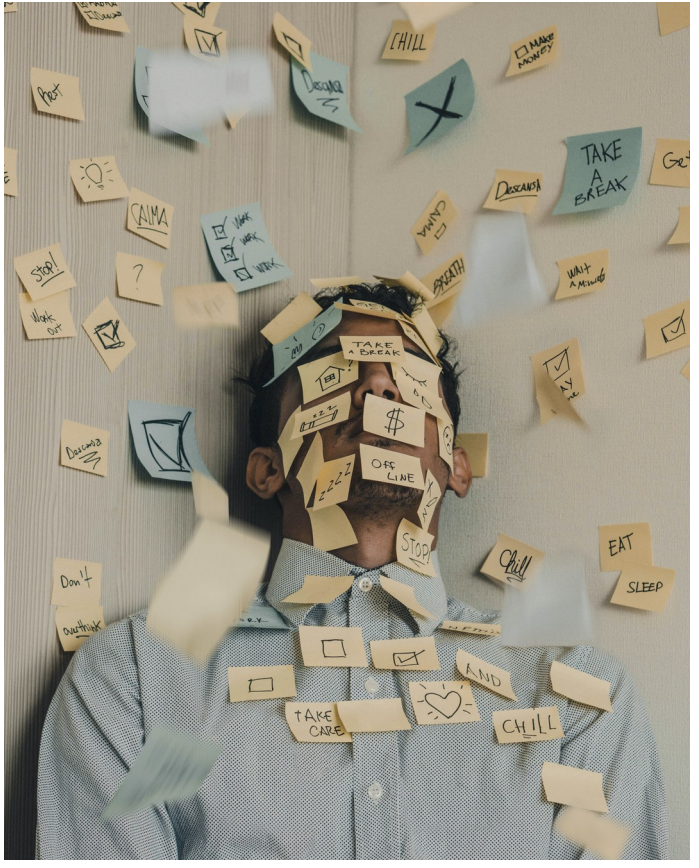
Manutenção de software...

Corretiva	Diagnosticar e corrigir erros
Adaptativa	Adaptar sistema a novo hardware
Perfectiva	Melhorar desempenho ou manutenibilidade

Reuso de software

Com o uso do software...

...surgem necessidades de melhoramentos e extensões



Reuso de software

Características da POO favorecem manutenção!

Modularidade

Facilita compreensão do efeito de mudanças no sistema

Polimorfismo

Herança

Reuso de software

Características da POO favorecem manutenção!

Modularidade

Facilita compreensão do efeito de mudanças no sistema

Polimorfismo

Reduz a quantidade de código a ser compreendido

Herança

Reuso de software

Características da POO favorecem manutenção!

Modularidade	Facilita compreensão do efeito de mudanças no sistema
Polimorfismo	Reduz a quantidade de código a ser compreendido
Herança	Permite criar novos códigos sem afetar os antigos

Reuso de software

Reuso não acontece por acaso!
Projetistas de software precisam...

Planejar o reuso de componentes antigos

Procurar novos componentes reutilizáveis

Reuso de software

Reescrever uma classe antiga para torná-la mais reusável é tão importante quanto inventar novas classes

Espera-se que programadores gastem tempo tanto quanto em ler código antigo e ver como reusá-lo, assim como para criar novos códigos

Reuso de software

A atitude mais relevante é a importância dada à criação de abstrações reusáveis.

Reuso de software

A atitude mais relevante é a importância dada à criação de abstrações reusáveis.

Abstrações úteis são geralmente criadas por programadores obcecados por simplicidade...

...que desejam reescrever código várias vezes para produzir classes fáceis de compreender e de especializar.

Reuso de software

Programadores que tornam suas soluções procedurais para um **particular problema** em uma **biblioteca genérica** são raros e valorosos.

Reuso de software

Abstrações úteis são geralmente projetadas de “baixo para cima” (são descobertas, e não inventadas)

Reuso de software

Abstrações úteis são geralmente projetadas de “baixo para cima” (são descobertas, e não inventadas)

Componentes gerais são criados resolvendo-se problemas específicos...

…e então reconhecendo que as soluções tem potencial de aplicabilidade maior.

Reusabilidade de software

“Designing reusable classes”

IFC Bnu - Prof. Hylson

Fotos de...

unsplash.com
linkedin, laputan.org
google (creative commons)