

Fundamentos Java e Orientação a Objetos



Por
Thiago Faria

5.2. Desafio: diagrama de classes

5. Orientação a Objetos - parte 2



Você é um analista de sistemas de um grande banco e surgiu uma demanda de um novo projeto para ser desenvolvido.

O usuário, da área de empréstimos e financiamentos, solicitou que fosse feito um sistema para simular financiamentos imobiliários para os clientes do banco.

Seu papel como analista será apenas de desenhar o diagrama de classes para depois o programador desenvolver o código-fonte, mas para isso, preste bastante atenção aos requisitos do usuário requisitante:

- Para que uma simulação de financiamento seja feita, deve-se ter em mãos os dados do imóvel e do cliente.
- Um imóvel possui as seguintes características: cidade, valor e estado (novo ou usado).
- Um cliente deve ter as seguintes características: nome, renda bruta e idade.

O algoritmo do financiamento não está em questão neste desafio, você precisa apenas desenhar a estrutura das classes em um diagrama de classes da UML.

Para facilitar, use uma ferramenta UML que preferir. Para Windows, recomendamos o [StarUML](#). Para Linux ou Mac, use [ArgoUML](#).

Se quiser mostrar o resultado de seu trabalho para os colegas do curso e para o instrutor, envie sua imagem para a web, torne-a disponível publicamente e escreva um comentário nesta página com o endereço da imagem. Um serviço gratuito para envio de imagens (não precisa nem de cadastro) é <http://troll.ws>.

Comentários sobre esta aula



Diogo Álvaro Bezerra - 22/03/2012 às 14:07
Segue meu Diagrama.

Link: <http://troll.ws/i/247nY1.jpg>



Normandes Júnior INSTRUTOR - 22/03/2012 às 14:14
É isso ai, muito bem.



Manuel Monteiro - 14/02/2012 às 09:28
Professor tem o ficheiro de iamgem no URL abaixo : <http://troll.ws/i/fuUy29.jpg>



Manuel Monteiro - 13/02/2012 às 12:55
Tenho duvida com relação os atributos do desafio



Normandes Júnior INSTRUTOR - 13/02/2012 às 18:23
Qual dúvida? Crie o projeto no StarUML e crie as classes que você acha que deve ter. Depois vá adicionando os atributos que achar necessário e me mande pra eu te ajudar, ok?



Manuel Monteiro - 15/02/2012 às 06:22
Normandes ,

ainda nao me respondeste com relação o desafio de UML .



Normandes Júnior INSTRUTOR - 17/02/2012 às 18:35
Manuel, olhei seu diagrama de classe e só tenho algumas considerações:
1º. Não acho necessário você colocar os atributos na classe Cliente como "nomeCliente", "rendaBrutaDeCliente" e "idadeCliente". Você pode usar simplesmente "nome", "rendaBruta" e "idade". Você já está na classe Cliente, não precisa que o atributo tenha esse nome.
2º. O Método de simulação não ficaria na classe Cliente, pense bem, um cliente saberia fazer essa simulação? Não, portanto acho melhor você criar uma classe Simulador que tenha um método "simular" que receba como parâmetro um Imóvel e um Cliente.
Altere e me mande para eu dar uma olhada.



Manuel Monteiro - 22/02/2012 às 08:01
Alterações efectuadas e podem verificar no link abaixo :
<http://troll.ws/i/64Ibdj.jpg>



Normandes Júnior INSTRUTOR - 22/02/2012 às 18:59
Já ficou melhor Manuel, porém ainda acho que o método simular que deveria receber os parâmetros e não ser um atributo da classe Simulador (repare que você colocou dois "a" no nome da classe).
O método ficaria assim:
simular(cliente: Cliente, imovel: Imovel): void

Repare também que no seu modelo está faltando os retornos dos métodos, lembre-se um método sempre tem retorno, nem que seja "void".



Manuel Monteiro - 23/02/2012 às 09:52
projecto actualizado. <http://troll.ws/i/YPIGGc.jpg>
Normandes estou a ter problema em termo de retorno dos metodos podes me dar uma ajuda sobre o assunto .



Normandes Júnior INSTRUTOR - 23/02/2012 às 18:23

Manuel assista mais uma vez a aula "3.11. Métodos com retorno". Lá explico detalhes sobre o retorno dos métodos. Seu diagrama de classe está melhor, o problema está nos métodos de Imóvel e Cliente. Para esse exemplo você não precisaria desses métodos. Um Imóvel não sabe se cadastrar e nem se consultar, você teria uma outra classe para fazer esse cadastro.

Compartilhe esta aula com seus amigos

[Twitter](#)[Facebook](#)

1. Introdução

- [1.1. Como aprender Java?](#) 5m 50s GRÁTIS
- [1.2. A história do Java](#) 2m 46s GRÁTIS
- [1.3. As plataformas Java e como elas evoluem](#) 10m 31s GRÁTIS
- [1.4. Máquina virtual Java](#) 8m 45s GRÁTIS
- [1.5. Baixando, instalando e configurando a JDK](#) 7m 59s GRÁTIS
- [1.6. Exercício: instalação da JDK](#) GRÁTIS

2. Fundamentos da linguagem

- [2.1. Codificando, compilando e executando o programa "oi mundo"](#) 13m 10s GRÁTIS
- [2.2. Exercício: codificando um primeiro programa](#) GRÁTIS
- [2.3. Comentários](#) 3m 3s GRÁTIS
- [2.4. Sequências de escape](#) 5m 14s GRÁTIS
- [2.5. Palavras reservadas](#) 3m 32s GRÁTIS
- [2.6. Convenções de código](#) 2m 28s GRÁTIS
- [2.7. Trabalhando com variáveis](#) 6m 18s GRÁTIS
- [2.8. Nomeando variáveis](#) 5m 42s GRÁTIS
- [2.9. Operadores aritméticos](#) 9m 36s GRÁTIS
- [2.10. Exercício: variáveis e operadores aritméticos](#) GRÁTIS
- [2.11. Tipos primitivos](#) 12m 0s GRÁTIS
- [2.12. Outros operadores de atribuição](#) 4m 43s GRÁTIS
- [2.13. Conversão de tipos primitivos](#) 12m 39s GRÁTIS
- [2.14. Promoção aritmética](#) 6m 25s GRÁTIS
- [2.15. Exercício: tipos primitivos e outros operadores de atribuição](#) GRÁTIS
- [2.16. Trabalhando com strings](#) 7m 5s GRÁTIS
- [2.17. Recebendo entrada de dados](#) 7m 41s GRÁTIS
- [2.18. Operadores de comparação e igualdade](#) 6m 40s GRÁTIS
- [2.19. Estruturas de controle if, else if e else](#) 12m 23s GRÁTIS
- [2.20. Exercício: Strings, entrada de dados, operadores de comparação e if else](#) GRÁTIS
- [2.21. Escopo de variáveis](#) 6m 3s GRÁTIS
- [2.22. Operadores lógicos](#) 15m 13s GRÁTIS
- [2.23. Exercício: operadores lógicos](#) GRÁTIS
- [2.24. Estrutura de controle switch](#) 7m 10s GRÁTIS
- [2.25. Operador ternário](#) 6m 49s GRÁTIS
- [2.26. Operadores de incremento e decremento](#) 8m 11s GRÁTIS
- [2.27. Estrutura de controle while](#) 5m 45s GRÁTIS
- [2.28. Estrutura de controle do-while](#) 3m 47s GRÁTIS
- [2.29. Estrutura de controle for](#) 4m 15s GRÁTIS
- [2.30. Cláusulas break e continue](#) 7m 2s GRÁTIS
- [2.31. Exercício: operador ternário, decremento e estruturas de repetição](#) GRÁTIS
- [2.32. Introdução e instalação do Eclipse IDE](#) 13m 40s GRÁTIS
- [2.33. Depurando códigos com o Eclipse](#) 8m 43s GRÁTIS
- [2.34. Exercício: instalando o Eclipse IDE](#) GRÁTIS

3. Orientação a Objetos - parte 1

- [3.1. O que é POO?](#) 2m 57s GRÁTIS
- [3.2. Classes e objetos](#) 5m 16s GRÁTIS
- [3.3. Criando uma classe com atributos](#) 2m 48s GRÁTIS
- [3.4. Instanciando objetos](#) 7m 59s GRÁTIS
- [3.5. Acessando atributos de objetos](#) 8m 32s GRÁTIS
- [3.6. Exercício: instanciando e acessando atributos do objeto](#) GRÁTIS
- [3.7. Composição de objetos](#) 9m 28s GRÁTIS
- [3.8. Valores padrão](#) 5m 59s GRÁTIS
- [3.9. Variáveis referenciam objetos](#) 9m 22s GRÁTIS
- [3.10. Criando, nomeando e chamando métodos](#) 8m 2s GRÁTIS
- [3.11. Métodos com retorno](#) 11m 13s GRÁTIS
- [3.12. Passando argumentos para métodos](#) 5m 25s GRÁTIS
- [3.13. Argumentos por valor ou referência](#) 7m 0s GRÁTIS
- [3.14. Exercício: composição de objetos e chamada de métodos](#) GRÁTIS

4. Wrappers, boxing e arrays

- [4.1. Wrappers do java.lang](#) 3m 31s GRÁTIS
- [4.2. Boxing](#) 6m 47s GRÁTIS
- [4.3. Desafio: wrappers e boxing](#) GRÁTIS
- [4.4. Trabalhando com arrays](#) 16m 37s GRÁTIS
- [4.5. Exercício: arrays](#) GRÁTIS

5. Orientação a Objetos - parte 2

- [5.1. Introdução à UML e diagrama de classes](#) 7m 31s GRÁTIS
- [5.2. Desafio: diagrama de classes](#) GRÁTIS
- [5.3. O objeto this](#) 8m 18s GRÁTIS
- [5.4. Construtores](#) 11m 43s GRÁTIS
- [5.5. Encapsulamento e modificadores de acesso public e private](#) 11m 7s GRÁTIS
- [5.6. Criando JavaBeans](#) 8m 40s GRÁTIS

- 5.7. Desafio: objeto this, construtores e JavaBeans

GRÁTIS
- 5.10. Modificadores static e final

12m 40s

GRÁTIS
- 5.13. Desafio: pacotes e enumerações

GRÁTIS
- 5.16. Sobreposição

7m 48s

GRÁTIS
- 5.19. Exercício: sobrecarga

GRÁTIS
- 5.22. Desafio: polimorfismo e classes abstratas

GRÁTIS
- 5.8. Organizando os projetos em pacotes

11m 51s

GRÁTIS
- 5.11. Desafio: static e final

GRÁTIS
- 5.14. Herança e modificador protected

10m 42s

GRÁTIS
- 5.17. Desafio: herança e sobreposição

GRÁTIS
- 5.20. Polimorfismo, casting de objetos e instanceof

18m 49s

GRÁTIS
- 5.23. Interfaces

11m 49s

GRÁTIS
- 5.9. Modificador de acesso default

6m 55s

GRÁTIS
- 5.12. Enumerações

17m 26s

GRÁTIS
- 5.15. Classe java.lang.Object

4m 13s

GRÁTIS
- 5.18. Sobrecarga

7m 48s

GRÁTIS
- 5.21. Classes abstratas

9m 49s

GRÁTIS
- 5.24. Exercício: interfaces e polimorfismo

GRÁTIS

6. Tópicos avançados

- 6.1. Coleta de lixo

8m 40s

GRÁTIS
- 6.2. Classe java.lang.Math

16m 6s

GRÁTIS
- 6.3. Desafio: classe java.lang.Math

GRÁTIS
- 6.4. Tratando e lançando exceções

29m 12s

GRÁTIS
- 6.5. Desafio: exceções

GRÁTIS
- 6.6. Classes String, StringBuffer e StringBuilder

8m 26s

GRÁTIS
- 6.7. Trabalhando com datas

19m 28s

GRÁTIS
- 6.8. Desafio: datas

GRÁTIS
- 6.9. Trabalhando com números

9m 12s

GRÁTIS
- 6.10. Desafio: números

GRÁTIS
- 6.11. Collections Framework

22m 25s

GRÁTIS
- 6.12. Desafio: collections

GRÁTIS
- 6.13. Arquivos JAR

6m 19s

GRÁTIS
- 6.14. Exercício: arquivos JAR

GRÁTIS
- 6.15. Documentação javadoc

9m 55s

GRÁTIS
- 6.16. Desafio: javadoc

GRÁTIS
- 6.17. Próximos passos

4m 8s

GRÁTIS
- 6.18. Conclusão

2m 6s

GRÁTIS

Cursos online

Depoimentos de alunos

Sobre nós

Cursos presenciais

Instrutores

Fale conosco

Apostilas gratuitas

Trabalhe conosco

AlgaWorks Softwares, Treinamentos e Serviços Ltda

Av. Afonso Pena, 3538, Átrio Business Center

CEP. 38400-710 - Uberlândia/MG - Brasil

Tel. +55 (34) 8400-6931 - comercial@algaworks.com