Programação e Análise Orientada a Objetos

Parte 1 - Introdução à Análise e Programação Orientada a Objetos

Mathias Santos de Brito

Universidade Estadual de Santa Cruz

Objetivos

- Entender a programação Orientada a Objetos e seus princípios.
- Entender a correlação direta entre Análise e Programação.

Orientação a Objetos: O que é?

- A orientação a objetos é um paradigma de programação, que tenta fazer com o ato de programar, seja mais próxima da realidade e do dia-a-dia das pessoas.

- Como o próprio nome já diz utilizamos o conceito de Objetos para programar.

Orientação a Objetos: O que é?

- A orientação a objetos foi criada para facilitar a programação de programas de computador, provendo:
  - Clareza: Códigos mais limpos e mais fáceis de entender.
  - Organização: Código bem divido e organizado, cada parte tendo uma função específica no programa.
  - Reuso: Promover a reutilização de código em outros programas que não o de origem.
  - Modularização: Facilitar a inserção de novas funcionalidades e desacoplar diferentes partes do programa que possuem diferentes funções.

Objetos: O que são?

- Os objetos são os componentes do nosso programa.
- Estamos acostumados a conviver e pensar em objetos no dia-a-dia, por exemplo, carro, televisão, diário de classe, computador.
- O que todos tem em comum? Todos possuem características, ou seja **atributos.** Esses atributos dizem respeito por exemplo, à cor de um carro, motorização deste carro, fabricante, etc.
- Objetos podem ser reais ou imaginários, existem muitos objetos que não podemos tocar, pegar, ou seja não existe fisicamente, mas que fazem parte do nosso dia-a-dia.

Objetos: O que são?

- Exemplos de objetos imaginários são, aula, locação, venda.
- Note que como acontece com os objetos reais, físicos, os objetos imaginários também possuem atributos.
- No caso de uma aula, teremos um professor, uma disciplina, uma sala associada.
- Todas essas informações são alvo de interesse para a escola e por tanto para um eventual programa de computador.

Objetos: O que são?

- Além de atributos, os objetos também interagem com alguém ou algo.
- No dia-a-dia efetuamos ações em diversos objetos com o intuito de que esse responda à nossa ação.
- Ao entrarmos em uma carro, a primeira ação que fazemos sobre esse objeto é **dar a partida**.
- Note que o objeto nos oferece os meios pelos quais podemos dar a partida, no caso a ignição.
- A estas interfaces, esta comunicação, ou ações possíveis de se efetuar em um objeto, damos o nome de **Métodos**.

Objetos: O que são?

- Pense **Atributos** como sendo características.
- Pense **Métodos** como sendo verbos, ações. Os métodos funcionam em partes como uma função. (Lembrem das funções em C).
- Estas são os dois principais conceitos por trás de um objeto.

Classes: O que são?

- É de extrema importância sabermos diferenciar uma **Classe** de um **Objeto**.
- Uma **Classe** é o molde do objeto, são os detalhes de como o Objeto será e como deverá funcionar.
- Um Objeto existe durante a execução do programa. Durante a execução criamos uma série de objetos baseados nas classes que foram definidas pelo programador.
- De fato é possível que falemos mais de Classes de que Objetos, daqui pra frente.

Classes: O que são?

- Em geral programamos/escrevemos uma **Classe**, que em tempo de execução poderá dar origem a um ou mais Objetos.
- Um Objeto existe, tem seu espaço reservado na memória e pode ter seus **Atributos** manipulados e **Métodos** invocados.
- Este objeto foi criado a partir de um molde, ou seja a **Classe**.

Análise OO: O que é?

- Ao iniciarmos um projeto para o desenvolvimento de um programa de computador, sem dúvidas devemos começar analisando o problema.

- Para dar suporte ao desenvolvimento de programas Orientados a Objetos, criou-se então métodos e procedimentos de análise completamente voltados para esta tecnologia.

- Para entender melhor, imagine que fluxogramas puro não ajudam a projetar um sistema Orientado a Objeto, ele é apenas uma ferramenta dentre muitas outras necessárias.

Análise OO: Pra que?

- Uma análise bem feita irá fazer com que o seu programa esteja o mais próximo possível do que o seu cliente deseja.
- Se bem executada ela irá inevitavelmente livrar o programador, ou a equipe de programadores de dores de cabeça futuras, como por exemplo:
  - Ter que modificar o código diversas vezes, depois de pronto, a pedido do cliente porque uma funcionalidade foi esquecida.
  - Problemas com manutenção e acréscimo de funcionalidade.
  - Dificuldade em reutilizar em outro programa o código escrito neste projeto.
  - Dentre outras...

Análise e Programação OO

- Obviamente que a Análise e a Programação se completam.
- Dizer quem deve ser feito primeiro é um tema complicado, existem diferentes abordagens para a Análise e o Desenvolvimento de um programa de computador.
- É natural pensar que a Análise deve ser feita primeiramente e a programação logo após.
- Vamos continuar utilizando esta abordagem, Análise primeiro, programação depois.
- Esta escolha não é pelo fato dessa abordagem ser a melhor, mas sim por ser a mais natural.

UML

- Para dar suporte à Análise Orientada a Objetos, foi criada uma série de conceitos e tipos de diagramas para dar suporte à Análise OO.

- Diversos empresas, pesquisadores e pessoas, desenvolviam suas próprias metodologias e diagramas, até que os três mais famosos resolveram unificar os seus métodos.

- Daí nasceu o que hoje é considerado um padrão no que diz respeito à Análise OO, a UML (Unified Modeling Language - Linguagem de Modelagem Unificada)

- Utilizaremos a UML para fazermos a análise dos nossos problemas, os conceitos serão introduzidos paralelamente com os conceitos da programação Orientada a Objetos.

Uma análise.

Análise na Prática

- Suponha que fomos contratados para desenvolver um programa de computador para uma Locadora de Vídeo. Obviamente iremos começar fazendo uma Análise.

- O Cliente então narra o que ele deseja:

- “Gostaria de um programa onde pudesse cadastrar meus clientes, registrar as locações e saber quem está com filmes em atraso. Além disso gostaria que fosse possível armazenar o máximo de informações possíveis sobre o filme para que o atendente pudesse encontrá-lo com base nas informações que o cliente possui como, Nome do Filme, Atores, Diretor, Distribuidora, dentre outras informações.”

Análise na Prática: Identificando Objetos

- Como estamos acostumados a pensar em coisas, objetos. A primeira coisa que tentamos fazer em uma Análise OO é identificar os objetos que existirão no programa.
- **Dica:** Os objetos serão sempre Sujeitos (no sentido gramatical da palavra).
- Então vamos começar a identificá-los. Vamos ver novamente o problema...

“Gostaria de um programa onde pudesse cadastrar meus clientes, registrar as locações e saber quem está com filmes em atraso. Além disso gostaria que fosse possível armazenar o máximo de informações possíveis sobre o filme para que o atendente pudesse encontrá-lo com base nas informações que o cliente possui como, Nome do Filme, Atores, Diretor, Distribuidora, dentre outras informações.”

Vamos identificar os Sujeitos na descrição do cliente?

“Gostaria de um programa onde pudesse cadastrar meus **clientes**, registrar as **locações** e saber quem está com **filmes** em atraso. Além disso gostaria que fosse possível armazenar o máximo de informações possíveis sobre o **filme** para que o **atendente** pudesse encontrá-lo com base nas informações que o **cliente** possui como, **Nome do Filme**, **Atores**, **Diretor**, **Distribuidora**, dentre outras informações.”

Análise na Prática: Objetos Identificados

- Objetos Reais, Físicos ou animados.
  - Cliente
  - Atendente
  - Filme
  - Ator
  - Diretor
- Objetos Imaginários ou Inanimados.
  - Locação

Análise na Prática: Modelando as Classes

- Uma vez que identificamos os objetos, devemos agora modelar estes objetos na forma de **Classes**, de uma forma gráfica e fácil de entender e visualizar.

Cliente

Atendente

Locação

Filme

Classes são representadas por um retângulo contendo o nome da Classe.

Ator

Diretor

Distribuidora

Análise na Prática: Identificando os Atributos da Classe

- Para cada Classe (o molde do Objeto) que criamos, teremos agora que identificar os atributos de cada um e acrescentarmos ao diagrama.
- Primeiro vamos voltar a analisar o diagrama com as classes e tentar identificar juntos, quais são os atributos de cada um deles.

Cliente

Atendente

Locação

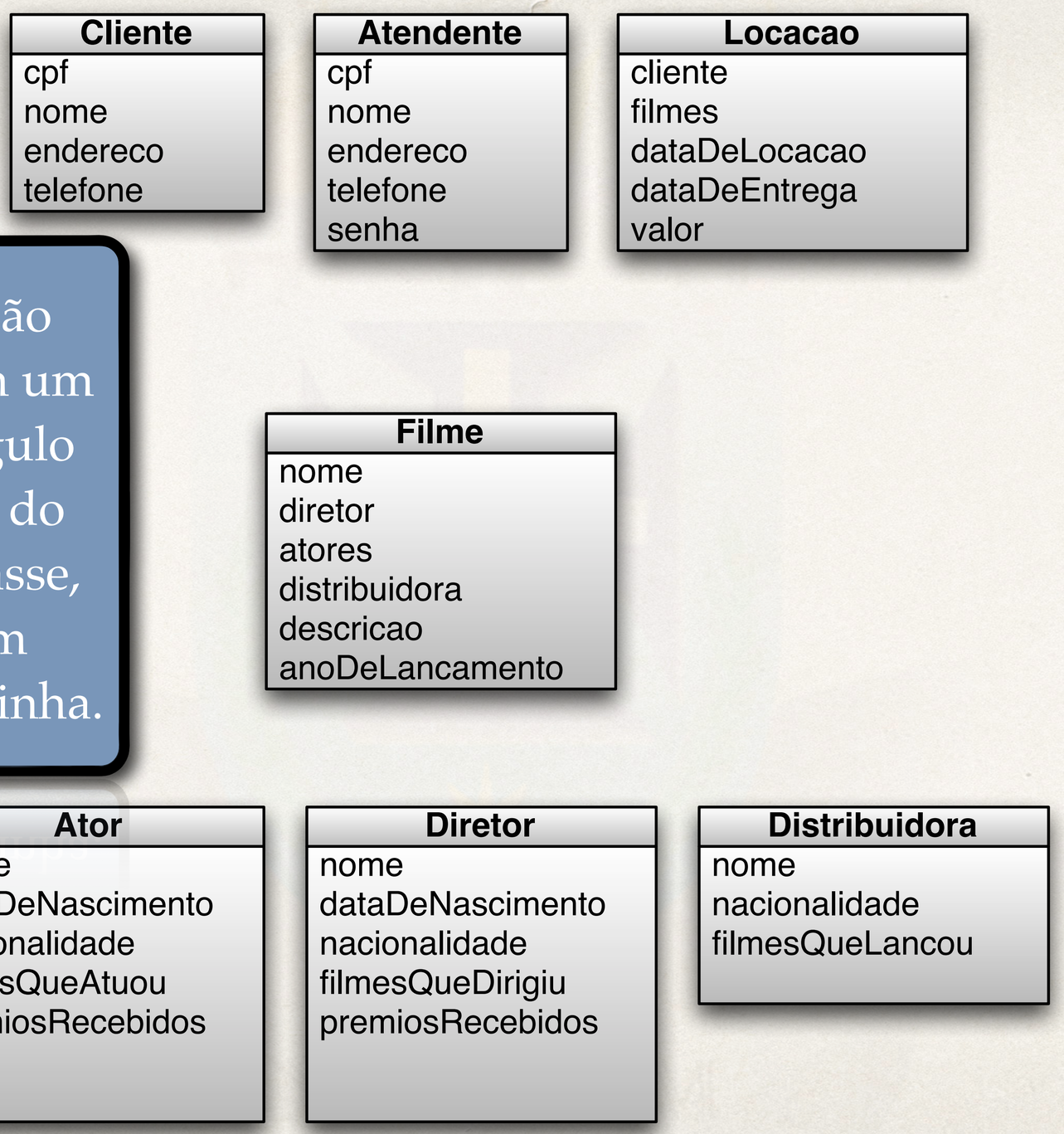
Filme

Ator

Diretor

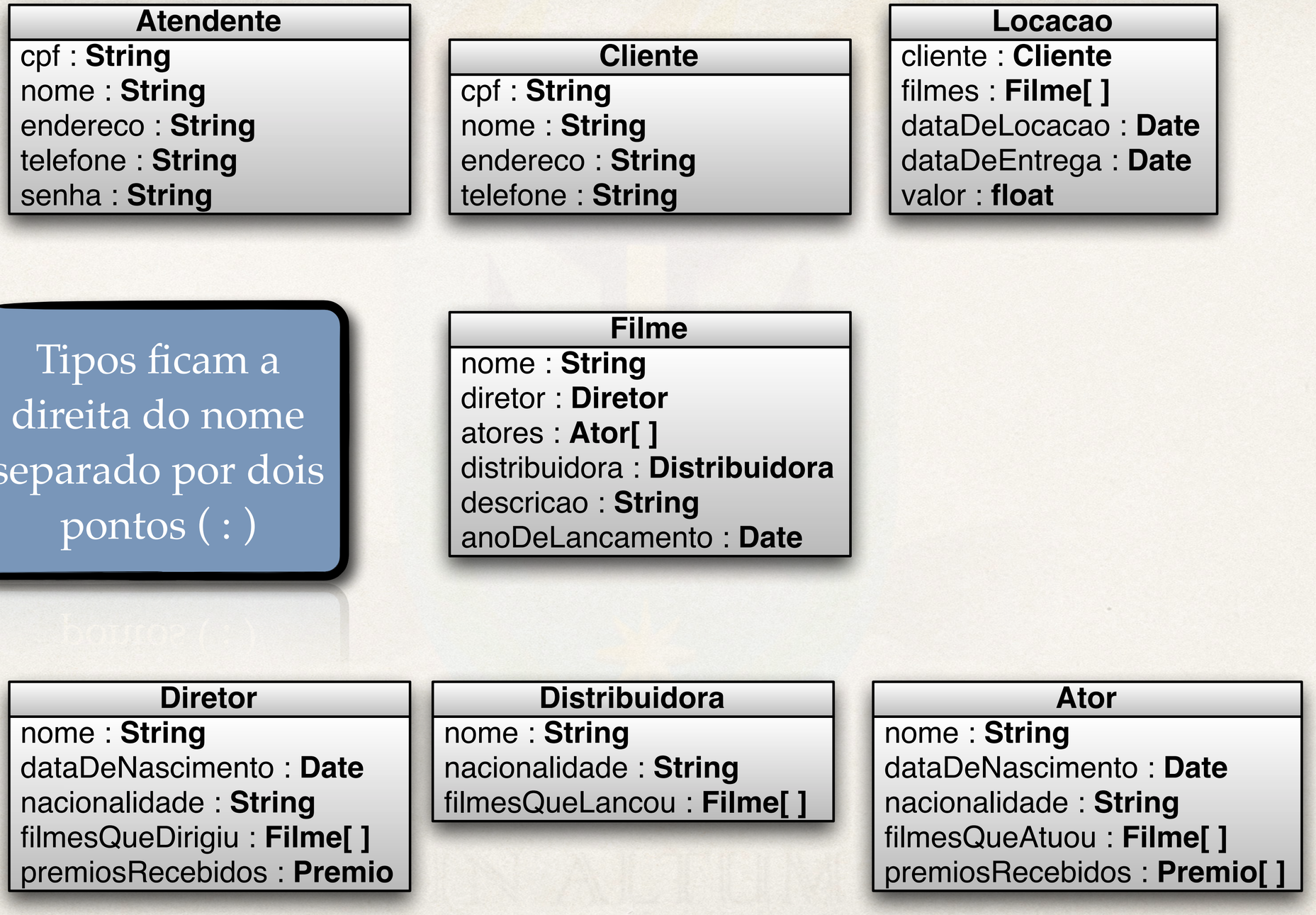
Distribuidora

Vamos ver como fica o nosso Diagrama de Classes com os Atributos


Cliente
cpf
nome
endereco
telefone
Atendente
cpf
nome
endereco
telefone
senha
Locacao
cliente
filmes
dataDeLocacao
dataDeEntrega
valor
Atributos são colocados em um outro retângulo logo abaixo do nome da Classe, sempre um atributo por linha.
Filme
nome
diretor
atores
distribuidora
descricao
anoDeLancamento
Ator
nome
dataDeNascimento
nacionalidade
filmesQueAtuou
premiosRecebidos
Diretor
nome
dataDeNascimento
nacionalidade
filmesQueDirigiu
premiosRecebidos
Distribuidora
nome
nacionalidade
filmesQueLancou

Análise na Prática: Identificando os Atributos da Classe

- Se tivermos um pouco de imaginação podemos comparar as Classes criadas aos **Structs** que criamos em C, certo?
- Um nome, e uma série de atributos que tem uma relação com um objeto ao qual este nome se refere.
- Se consegue visualizar isso é ótimo, deve ter percebido que falta o tipo do Atributo certo, vamos então colocá-los.


Atendente
cpf : String
nome : String
endereco : String
telefone : String
senha : String
Cliente
cpf : String
nome : String
endereco : String
telefone : String
Locacao
cliente : Cliente
filmes : Filme[ ]
dataDeLocacao : Date
dataDeEntrega : Date
valor : float
Tipos ficam a direita do nome separado por dois pontos (:)
Filme
nome : String
diretor : Diretor
atores : Ator[ ]
distribuidora : Distribuidora
descricao : String
anoDeLancamento : Date
Diretor
nome : String
dataDeNascimento : Date
nacionalidade : String
filmesQueDirigiu : Filme[ ]
premiosRecebidos : Premio
Distribuidora
nome : String
nacionalidade : String
filmesQueLancou : Filme[ ]
Ator
nome : String
dataDeNascimento : Date
nacionalidade : String
filmesQueAtuou : Filme[ ]
premiosRecebidos : Premio[ ]

Análise na Prática: Identificando os Atributos da Classe

- Notou que Classes são **tipos definidos pelo usuário.**
- Veja que declaramos atributos cujos tipos são Classes que definimos.
- Note também que definimos vetores de determinados tipos, no caso do filme, ele possui mais de um Ator, portanto tivemos que definir um vetor para que possamos armazenar vários atores.
- Mantenha isso em mente iremos entrar em mais detalhes depois.

Uma parada para falarmos sobre nomeação de Classes, Atributos e Métodos.

Análise na Prática: Nomeando classes e atributos

- Como pode perceber os nomes que demos às nossas no segundo diagrama já não possui caracteres especiais, nem espaços.
- Por convenção nomes de classes devem possuir a primeira letra maiúscula.
- A nomeação de classes, atributos e métodos são feitas utilizando-se uma notação conhecida como **“Notação Camelo”.**
- A Notação Camelo diz que em nomes de classes, atributos e métodos com mais de uma palavra não se usa espaços, nem sublinhados.
- Ao invés disso **a primeira letra de cada palavra deve ser maiúscula.**

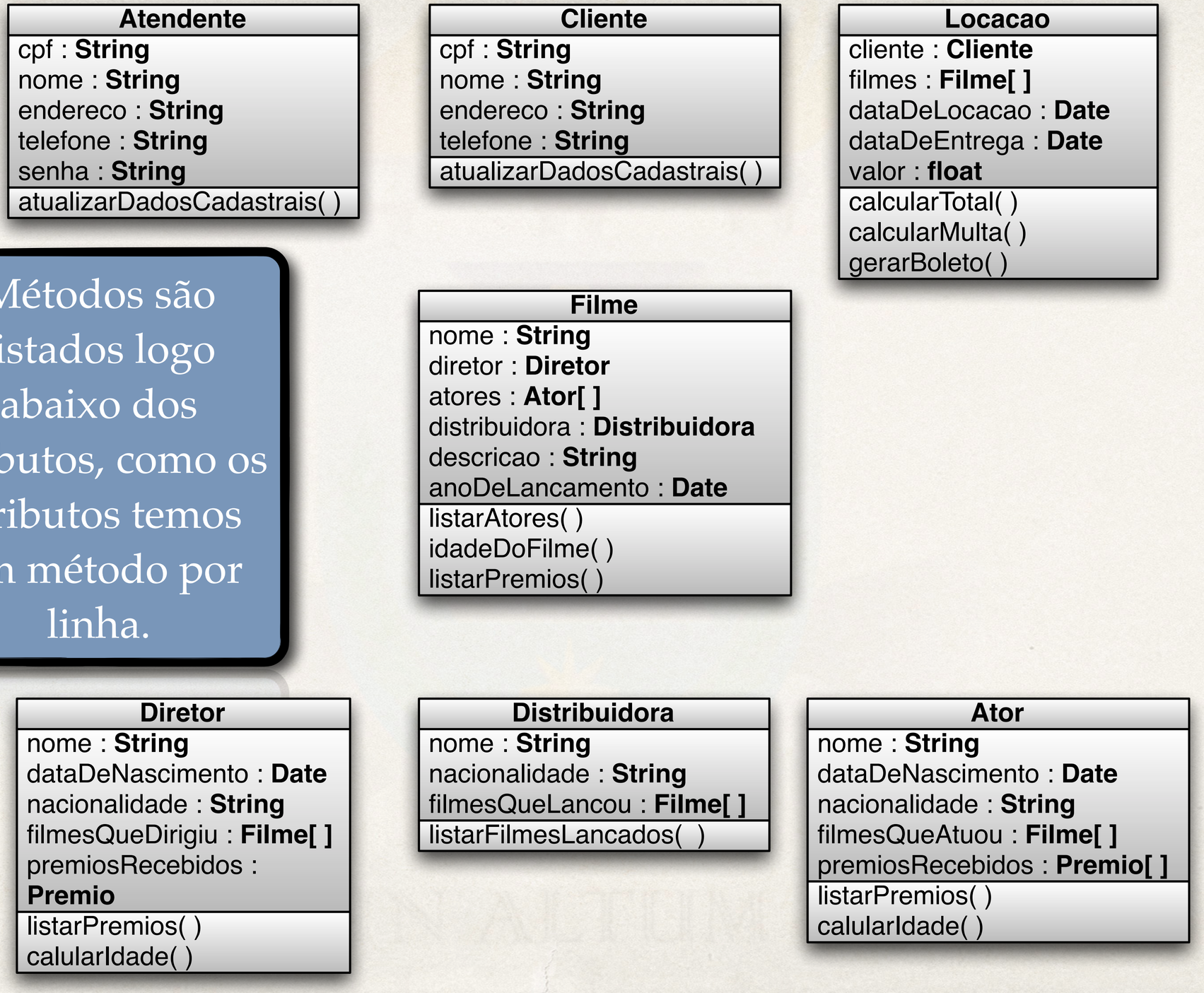
Análise na Prática: Nomeando classes e atributos

- Alguns exemplos observados no diagrama anterior são:
  - dataDeNascimento
  - filmesQueAtuou
  - anoDeLancamento
  - dataDeEntrega

De volta à Análise: Identificando os Métodos

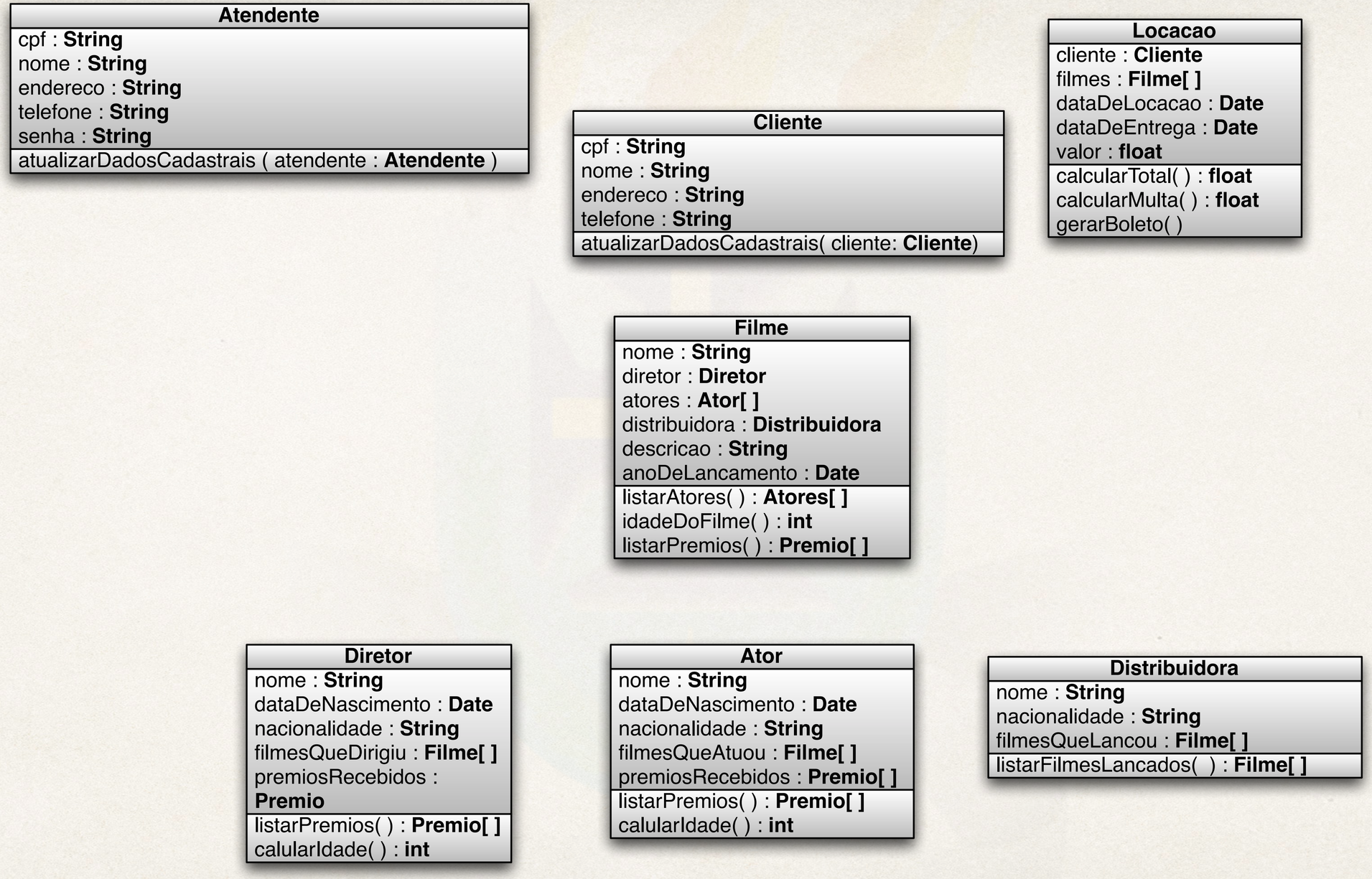
Análise na Prática: Identificando Métodos

- Nesta etapa iremos tentar detectar quais são as operações que uma Classe oferece pra nós, quais são as ações que podemos realizar em um objeto, **os métodos**.
- Vamos tentar colocar as ações no nosso diagrama?
- Observem como o diagrama está agora e tente identificar alguns métodos.


Atendente
cpf : String
nome : String
endereco : String
telefone : String
senha : String
atualizarDadosCadastrais( )
Cliente
cpf : String
nome : String
endereco : String
telefone : String
atualizarDadosCadastrais( )
Locacao
cliente : Cliente
filmes : Filme[ ]
dataDeLocacao : Date
dataDeEntrega : Date
valor : float
calcularTotal( )
calcularMulta( )
gerarBoleto( )
Métodos são listados logo abaixo dos atributos, como os atributos temos um método por linha.
Filme
nome : String
diretor : Diretor
atores : Ator[ ]
distribuidora : Distribuidora
descricao : String
anoDeLancamento : Date
listarAtores( )
idadeDoFilme( )
listarPremios( )
Diretor
nome : String
dataDeNascimento : Date
nacionalidade : String
filmesQueDirigiu : Filme[ ]
premiosRecebidos : Premio
listarPremios( )
calularIdade( )
Distribuidora
nome : String
nacionalidade : String
filmesQueLancou : Filme[ ]
listarFilmesLancados( )
Ator
nome : String
dataDeNascimento : Date
nacionalidade : String
filmesQueAtuou : Filme[ ]
premiosRecebidos : Premio[ ]
listarPremios( )
calularIdade( )

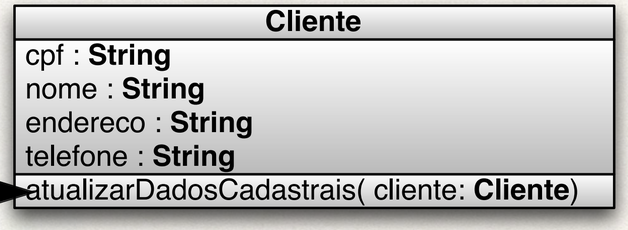
Análise na Prática: Identificando Métodos

- A maneira pela qual escrevi os métodos nos lembram as funções em C, certo?
- E isso é verdade, portando para que ele se torne ainda mais parecido precisamos colocar os tipos de retorno, bem como os parâmetros que o método recebe.
- Vamos conferir?

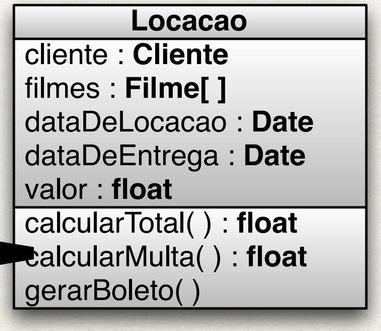

Atendente
cpf : String
nome : String
endereco : String
telefone : String
senha : String
atualizarDadosCadastrais (atendente : Atendente)
Cliente
cpf : String
nome : String
endereco : String
telefone : String
atualizarDadosCadastrais(cliente: Cliente)
Locacao
cliente : Cliente
filmes : Filme[ ]
dataDeLocacao : Date
dataDeEntrega : Date
valor : float
calcularTotal( ) : float
calcularMulta( ) : float
gerarBoleto( )
Filme
nome : String
diretor : Diretor
atores : Ator[ ]
distribuidora : Distribuidora
descricao : String
anoDeLancamento : Date
listarAtores( ) : Atores[ ]
idadeDoFilme( ) : int
listarPremios( ) : Premio[ ]
Diretor
nome : String
dataDeNascimento : Date
nacionalidade : String
filmesQueDirigiu : Filme[ ]
premiosRecebidos : Premio
listarPremios( ) : Premio[ ]
calularIdade( ) : int
Ator
nome : String
dataDeNascimento : Date
nacionalidade : String
filmesQueAtuou : Filme[ ]
premiosRecebidos : Premio[ ]
listarPremios( ) : Premio[ ]
calularIdade( ) : int
Distribuidora
nome : String
nacionalidade : String
filmesQueLancou : Filme[ ]
listarFilmesLancados( ) : Filme[ ]

Análise na Prática: Identificando Métodos

Observe que os parâmetros recebidos pelo método são identificados dentro dos parêntese. Sempre primeiro o nome da variável : e depois o tipo.



O tipo de retorno é colocado depois do símbolo : (dois pontos) assim como em uma variável, **o tipo de retorno não tem nome.**



Exercício

- Agora que vimos uma boa parte da construção de um diagrama de Classes e hora de praticar os conhecimentos adquiridos. Faça o mesmo procedimento para o seguinte problema..

- “Uma empresa deseja um programa para controlar o estoque. Ela precisa que um funcionário autorizado, dê entrada dos produtos, bem como a saída. Os produtos são divididos em categorias, e cada categoria tem como destino um departamento diferente da empresa. O funcionário que irá retirar o produto deve ter autorização do departamento que eles está associado.”

- Baseado nas informações acima crie um diagrama de classes como o visto na sala de aula.