Metodologias de Aprendizado

Diagrama de Classe

Prof. Rafael Ris-Ala

Metodologias de Aprendizado

Diagrama de Classe

Prof. Rafael Ris-Ala

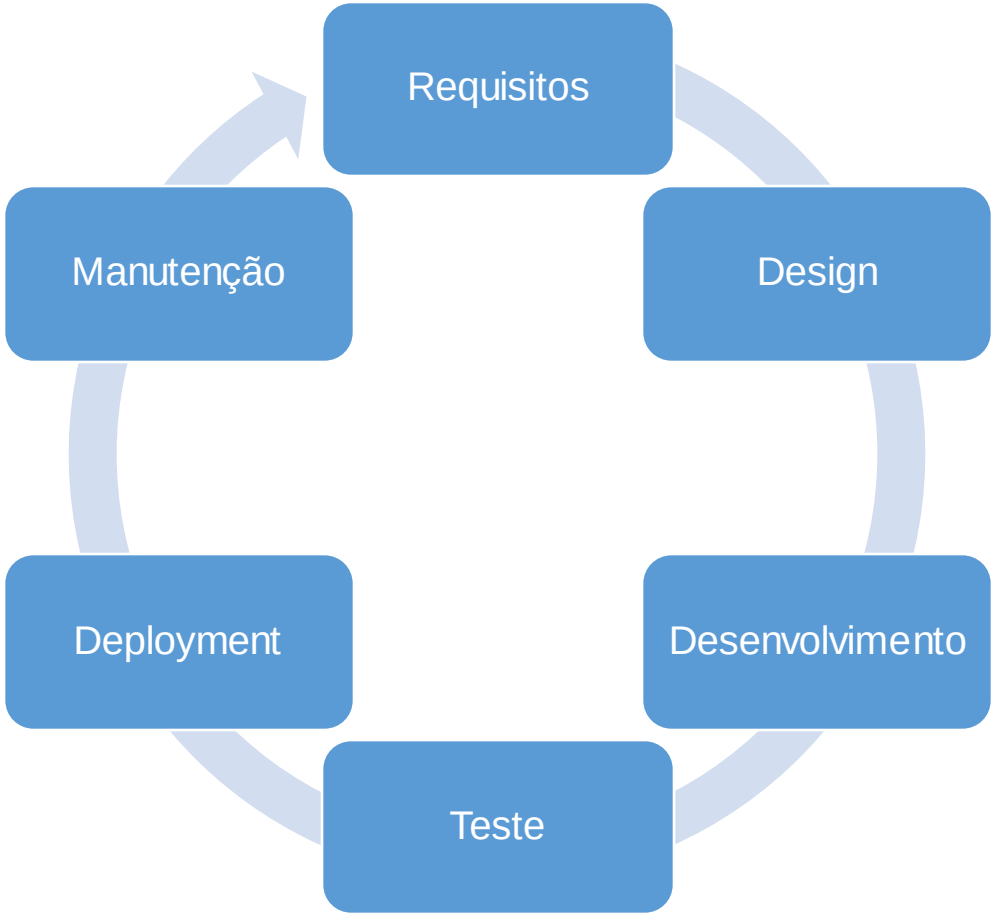
Nesta aula

- Conceito.
- Representação.
- Desenvolvimento.
- Resumo.

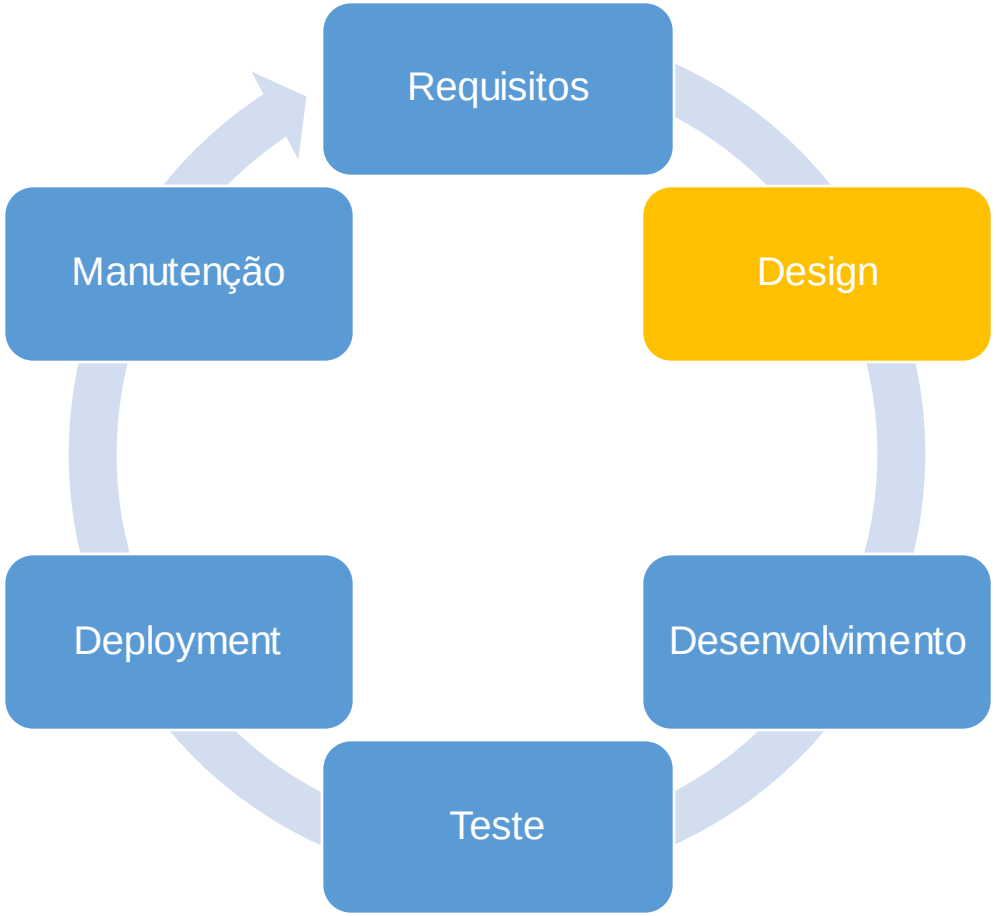
Nesta aula

- **Conceito.**
- Representação.
- Desenvolvimento.
- Resumo.

Desenvolvimento de Software

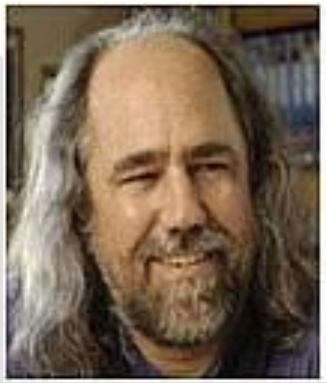


Desenvolvimento de Software



Origem do UML

- **Booch, Jacobson e Rumbaugh na Rational Software (1994).**
- **UML 1.0 (1997).**
- **OMG (2005).**



Conceito

- Linguagem de Modelagem Unificada (UML):
  - Linguagem padronizada para documentar e modelar aplicações de software.
  - Visualizar a arquitetura de um sistema por meio de diagramas.
  - Independente de linguagem.
- Auxilia a comunicação para o Desenvolvimento de Software.
- Diagrama: representação gráfica do modelo de um sistema.

Diagramas UML

- Estruturais
  - Classes
  - Objetos
  - Pacotes
  - ...
- Comportamentais
  - Caso de Uso
  - Sequência
  - Máquina de Estados
  - Atividade
  - ...

Diagrama de Classe UML

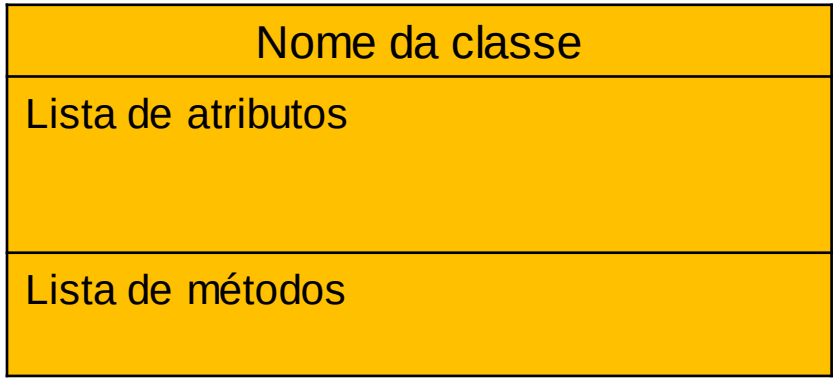
- Utilizado para descrever a estrutura estática de classes de um sistema.
- Permite definir: atributos, métodos (operações) e relacionamentos entre as classes.
- Diagrama mais popular.
- Descreve o que deve estar no sistema modelado.

Notação da Classe em UML

- Organização em 3 componentes:
  - Nome: identificação da classe;
  - Atributos: lista de atributo.

    Ex.: nome: tipo
  - Métodos: lista de operações.

    Ex.: método(parâmetros): retorno
- Visibilidade:
  - - Privado
  - + Público

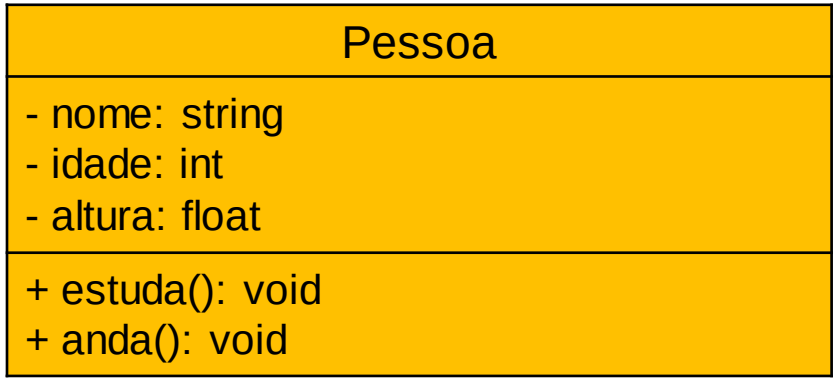


Notação da Classe em UML

- Organização em 3 componentes:
  - Nome: identificação da classe;
  - Atributos: lista de atributo.

    Ex.: nome: tipo
  - Métodos: lista de operações.

    Ex.: método(parâmetros): retorno
- Visibilidade:
  - - Privado
  - + Público



Exemplo

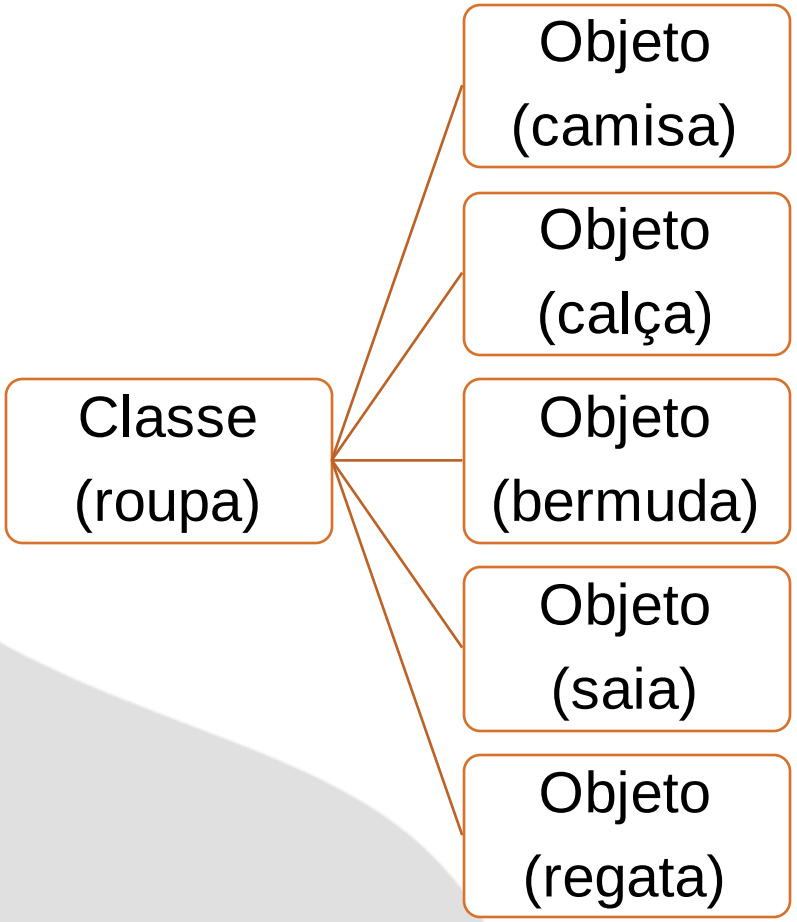
- **Classe**: Pessoa
  - **Atributos**: Nome, idade, altura...
  - **Métodos**: Estudar, andar...

Exemplo

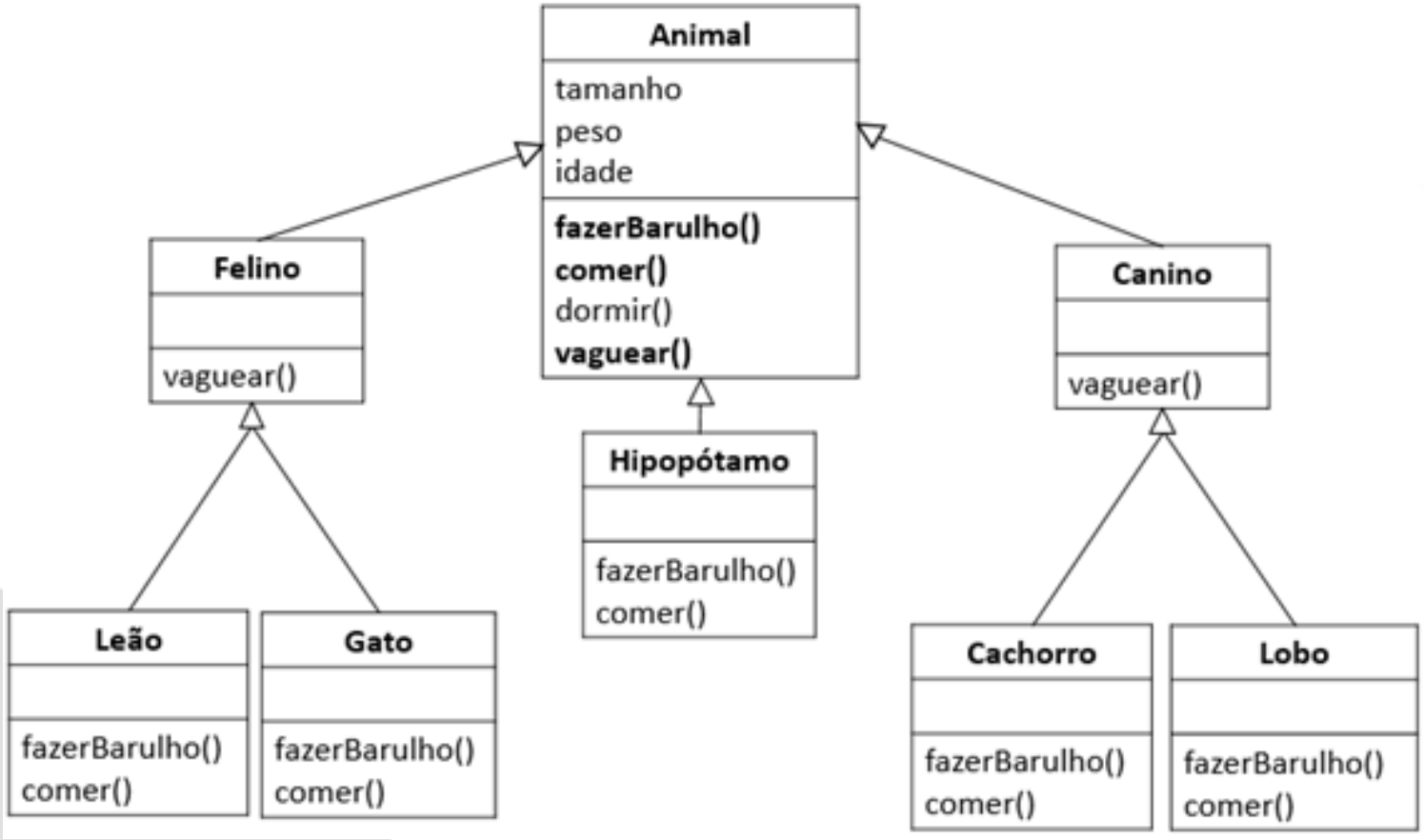
- **Classe**: Pessoa
  - **Atributos**: Nome, idade, altura...
  - **Métodos**: Estudar, andar...

Instância	Nome	Idade	Altura
instancia1	Rafael	39	1,87 m
instancia2	Pedro	18	1,72 m
instancia3	Carlos	32	1,81 m
instancia4	Lucas	27	1,79 m

Molde e instância



Herança de Classe



Boas Práticas

- Nome da classe deve ser significativo.
- Identificar o relacionamento entre os elementos.
- Especificar os atributos e métodos de cada classe.

Benefícios do Diagrama de Classe

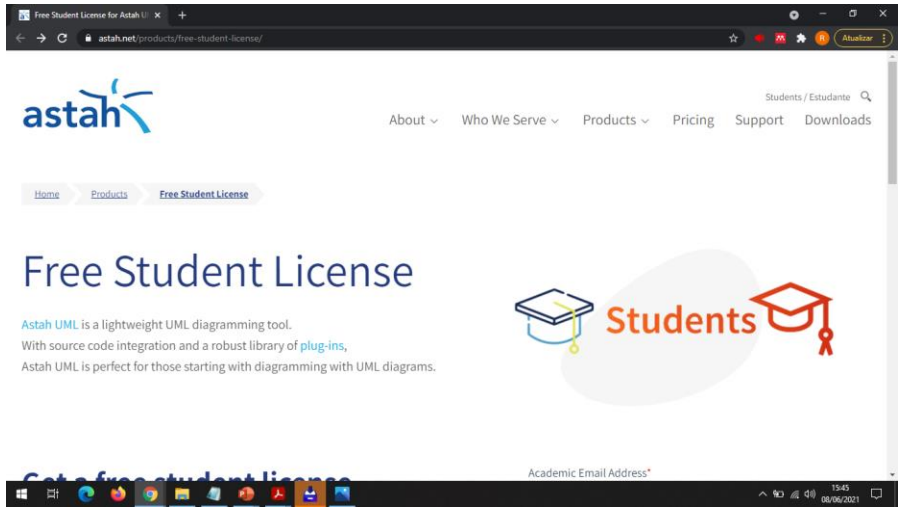
1. Ilustra modelos de dados, não importa quão simples ou complexo.
2. Expressa visualmente as necessidades de um sistema.
3. Entende melhor a visão dos esquemas de uma aplicação.
4. Cria gráficos que detalham o código a ser programado.
5. Fornece uma descrição independente da implementação.
6. Facilita a comunicação entre *stakeholders* e desenvolvedores.

Nesta aula

- Conceito.
- **Representação.**
- Desenvolvimento.
- Resumo.

Prática da Diagramação de Classe

- Astah. Versão 8.3:
- https://astah.net/products/free-student-license/
- Licença para estudante:
- https://astah.net/support/set-student-license/



Astah - [no_title]

File Edit Diagram Alignment View Tools Window Plugin Help

- Class Diagram
- UseCase Diagram
- Statemachine Diagram
- Activity Diagram
- Sequence Diagram
- Communication Diagram
- Component Diagram
- Deployment Diagram
- Composite Structure Diagram
- [Pro]Flowchart
- [Pro]Data Flow Diagram
- [Pro]ER Diagram
- [Pro]CRUD
- Mindmap
- [Pro]Requirement Diagram
- [Pro]Requirement Table

Structure

Map

No Selection

22:27
09/06/2021

Astah - [C:\Users\Rafael\Desktop\2021-IGTI\Diagrama de Classe\Class_2.asta] (*)

File Edit Diagram Alignment View Tools Window Plugin Help

Structure Inheritance Diagram Search

- Class_2
  - Class Diagram0
  - date
  - Localidade
  - Pessoa

Map

Template Parameter Constraint Language Hyperlink
Dependency Association Property
Base Stereotype Attribute Operation Generalization

Namespace

Name	Pessoa
Visibility	public
Abstract	false
Leaf	false
Active	false

Definition

Class Diagram0

Class Diagram0 / Class Diagram

pkg

Pessoa

16:12
09/06/2021

Astah - [C:\Users\Rafael\Desktop\2021-IGTI\Diagrama de Classe\Class_2.asta] (*)

File Edit Diagram Alignment View Tools Window Plugin Help

Structure Inheritance Diagram Search

- Class_2
  - Class Diagram0
  - date
  - Localidade
  - Pessoa

Map

Template Parameter Constraint Language Hyperlink
Dependency Association Property
Base Stereotype Attribute Operation Generalization

Namespace

Name	Localidade
Visibility	public
Abstract	false
Leaf	false
Active	false

Definition

Class Diagram0

Class Diagram0 / Class Diagram

pkg

Pessoa

- nome : string
- idade : int

+ estuda() : void
+ anda(destino : string) : void

16:11
09/06/2021

Astah - [C:\Users\Rafael\Desktop\2021-IGTI\Diagrama de Classe\Class_2.asta] (*)

File Edit Diagram Alignment View Tools Window Plugin Help

Structure Inheritance Diagram Search

- Class_2
  - Class Diagram0
  - date
  - Localidade
  - Pessoa
    - nome:string
    - idade:int
    - estuda
    - anda
    - :Localidade

Map

Base Hyperlink Initial Visibility

Namespace

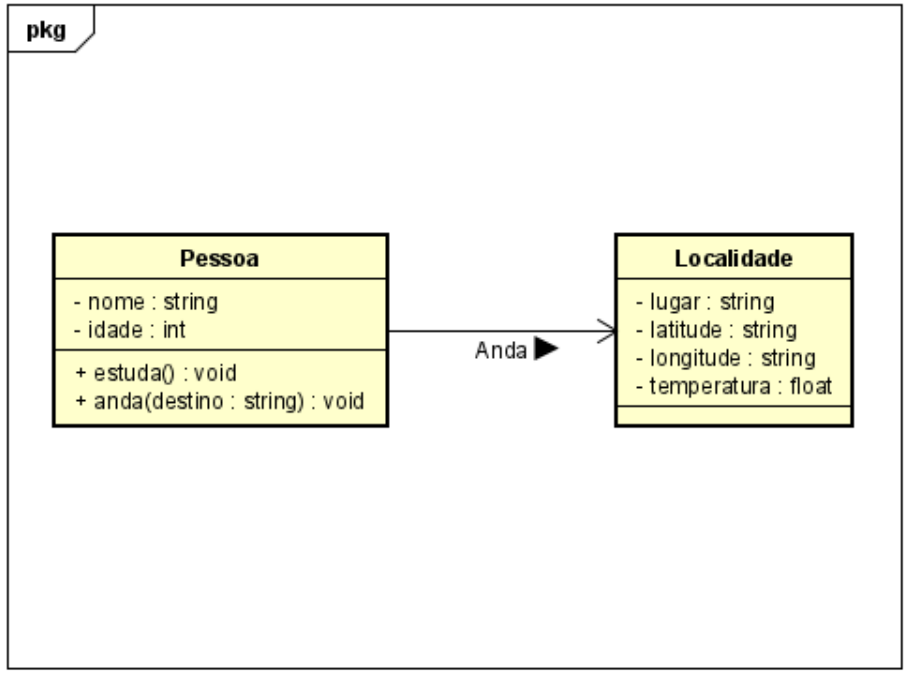
Name: Class Diagram0

Frame Visibility

Definition

Class Diagram0

Class Diagram0 / Class Diagram



16:14
09/06/2021

Astah - [C:\Users\Rafael\Desktop\2021-IGTI\Diagrama de Classe\Class_2.asta] (*)

File Edit Diagram Alignment View Tools Window Plugin Help

Structure Inheritance Diagram Search

- Class_2
  - Class Diagram0
  - date
  - Localidade
  - Pessoa

Map

Base Hyperlink Initial Visibility

Namespace

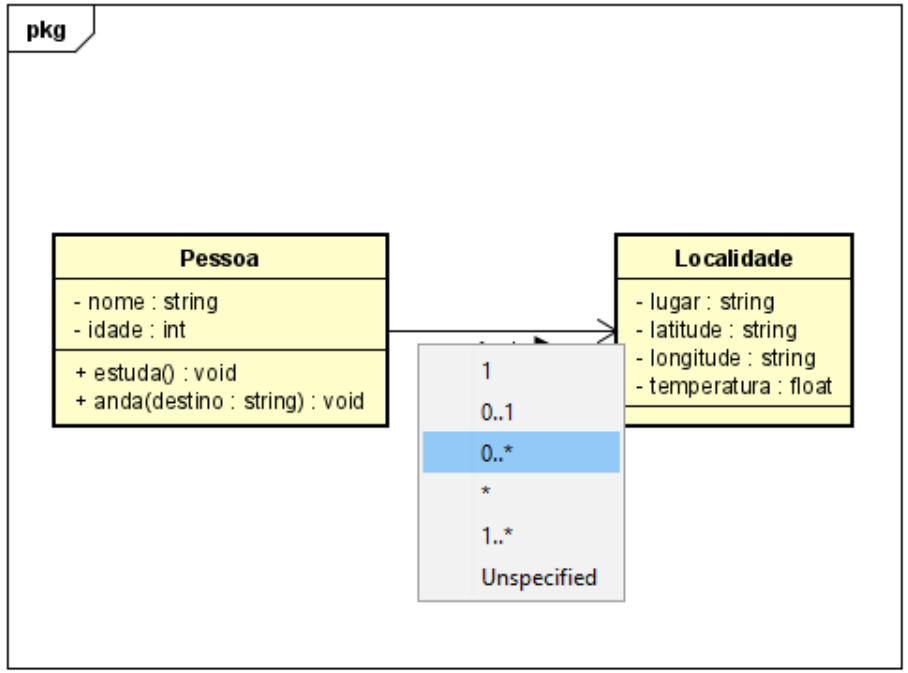
Name: Class Diagram0

Frame Visibility

Definition

Class Diagram0

Class Diagram0 / Class Diagram



16:11
09/06/2021

Astah - [C:\Users\Rafael\Desktop\2021-IGTI\Diagrama de Classe\Class_2.asta] (*)

File Edit Diagram Alignment View Tools Window Plugin Help

Structure Inheritance Diagram Search

- Class_2
  - Class Diagram0
  - date
  - Localidade
  - Pessoa

Map

Base Hyperlink Initial Visibility

Namespace

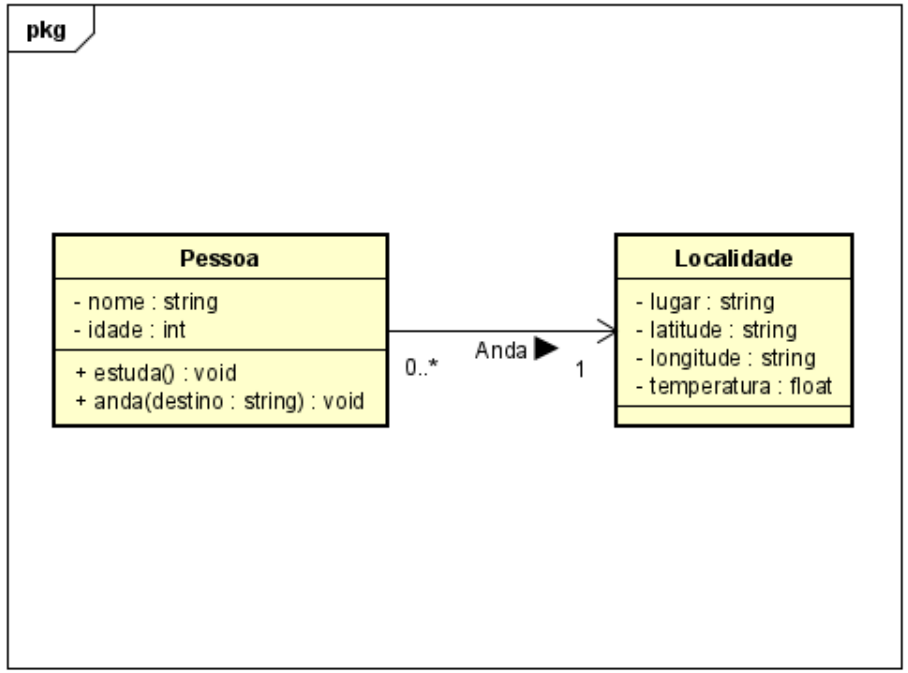
Name: Class Diagram0

Frame Visibility

Definition

Class Diagram0

Class Diagram0 / Class Diagram

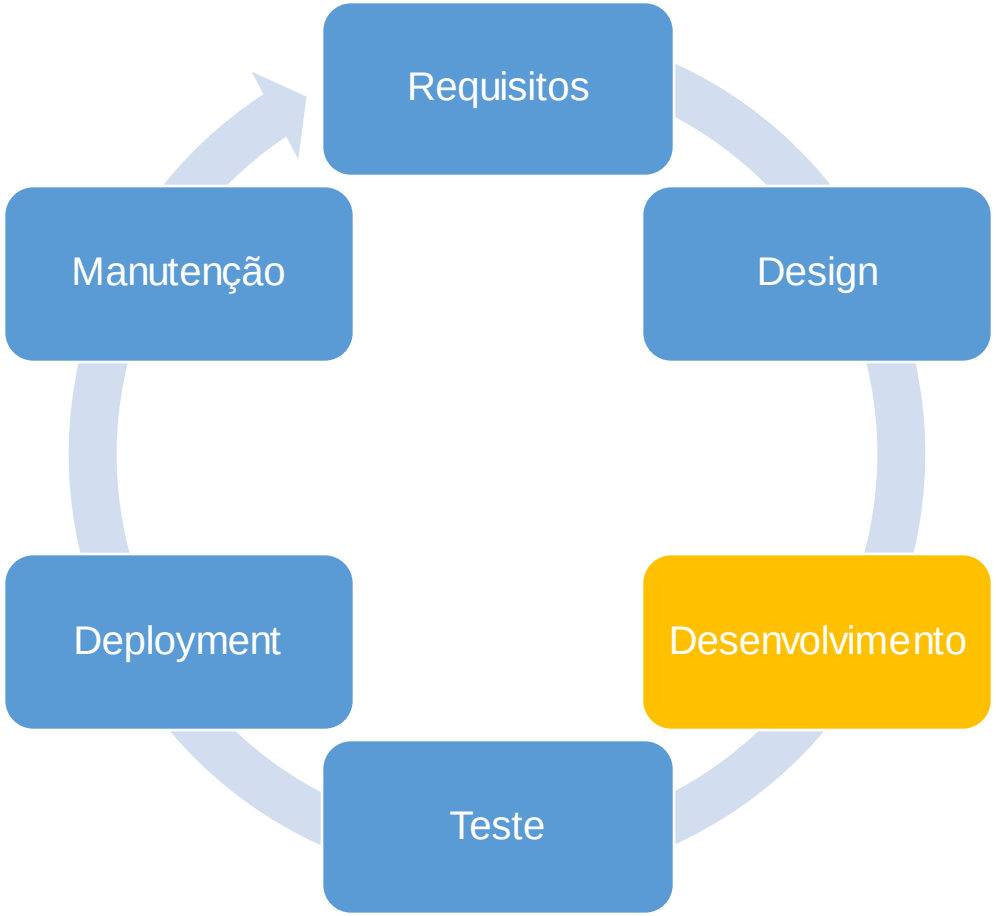


16:10
09/06/2021

Nesta aula

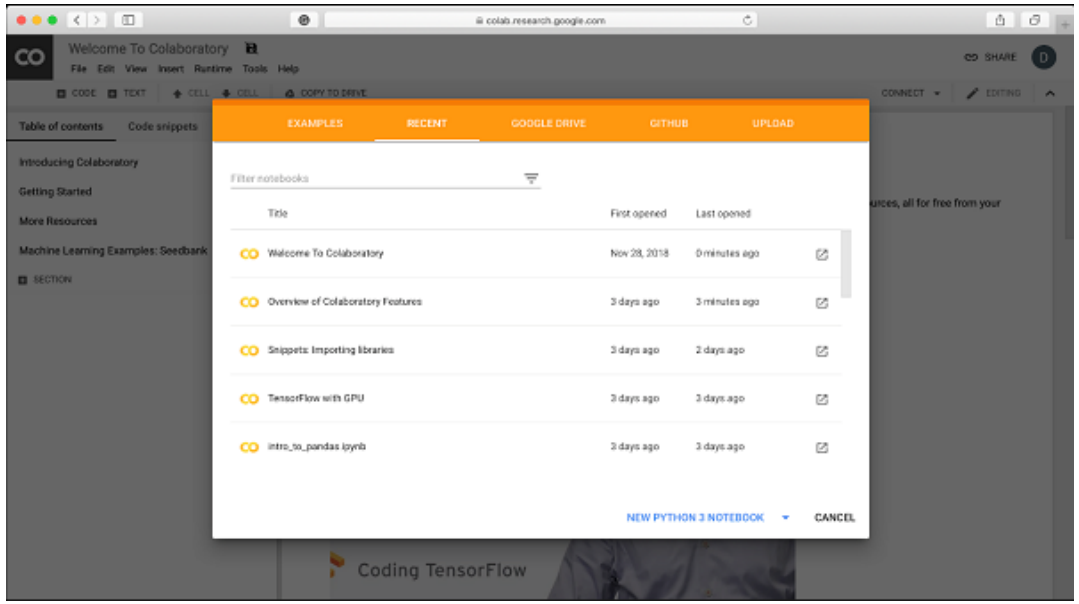
- Conceito.
- Representação.
- **Desenvolvimento.**
- Resumo.

Desenvolvimento de Software



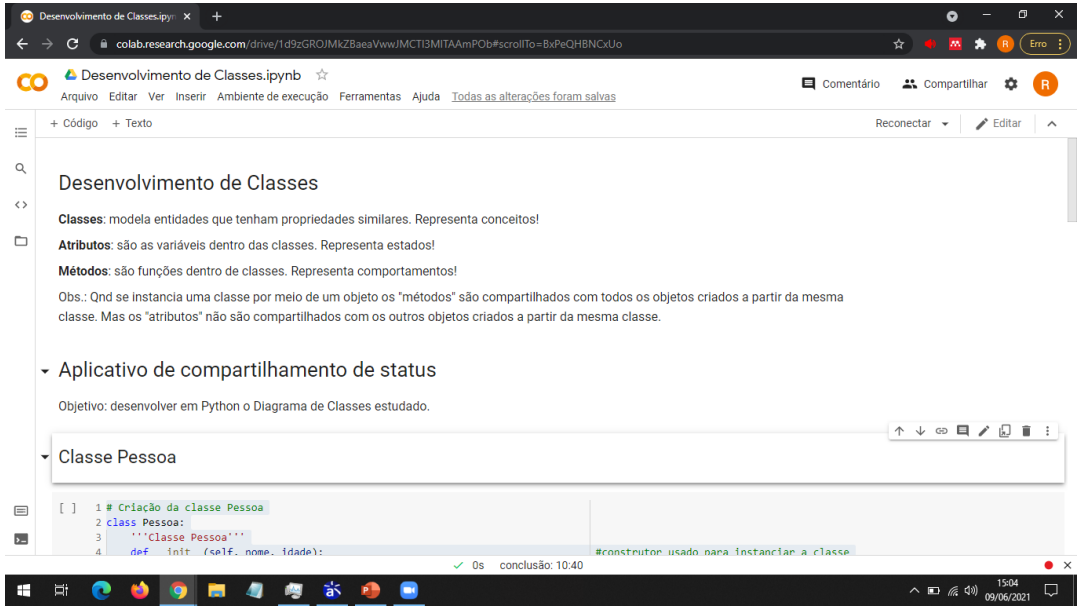
Prática com Código

- Ambiente de desenvolvimento Google Colab:
- https://colab.research.google.com/



Prática com Código

- Desenvolvimento em Python:
- https://colab.research.google.com/drive/1d9zGROJMkZBaeaVwwJMCTI3MlTAAmPOb?usp=sharing



Sintaxe

```
# Sintaxe de uma classe:
class NomeClasse:                   # Cabeçalho
    '''docstring '''                # Comentário sobre a classe
    dado = valor                    # Atributo da classe
    def metodo_construtor           # Método construtor. Toda classe deve ter um!
        codigo metodo construtor    #
    def metodo(self, ...)           # Método geral. self permite acessar as propriedades de uma instância
        self.membro = valor         # Dado de instância (self)
```

Exemplo:

Classe Pessoa

```
[ ]
1 # Criação da classe Pessoa
2 class Pessoa:
3     '''Classe Pessoa'''
4     def __init__(self, nome, idade):
5         self.nome = nome
6         self.idade = idade
7
8     def Estuda(self):
9         print(self.nome + " está estudando...")
10
11    def Anda(self, destino):
12        print(self.nome + " está andando até " + destino)
```

Exemplo:

Classe Pessoa

```
[ ]
```

```python
# Criação da classe Pessoa
class Pessoa:
    '''Classe Pessoa'''
    def __init__(self, nome, idade):
        self.nome = nome
        self.idade = idade

    def Estuda(self):
        print(self.nome + " está estudando...")

    def Anda(self, destino):
        print(self.nome + " está andando até " + destino)
```

```
[ ]
```

```python
# Definição de uma instância:
pedro = Pessoa('Pedro', 18)
# Testando a instância acessando o atributo:
print(pedro.idade)
# Testando a instância chamando o método:
pedro.Estuda()
```

```
18
Pedro está estudando...
```

Exemplo:

Classe Localidade

```
[ ]  1 # Criação da classe Localidade:
     2 class Localidade:
     3     '''Classe Localidade'''
     4     def __init__(self, lugar, latitude, longitude, temperatura):
     5         self.lugar = lugar
     6         self.latitude = latitude
     7         self.longitude = longitude
     8         self.temperatura = temperatura
```

Exemplo:

Classe Localidade

```
[ ]  1 # Criação da classe Localidade:
     2 class Localidade:
     3     '''Classe Localidade'''
     4     def __init__(self, lugar, latitude, longitude, temperatura):
     5         self.lugar = lugar
     6         self.latitude = latitude
     7         self.longitude = longitude
     8         self.temperatura = temperatura
```

```
[ ]  1 # Definição de uma instância:
     2 praia = Localidade('Praia de Ipanema', '-22.9836210655', '-43.2027158558', 42.8)
     3 # Testando a instância acessando o atributo:
     4 print(praia.lugar)
```

```
Praia de Ipanema
```

Exemplo:

Teste de uma classe chamando a outra

```
1 # Testando a instância chamando o método e passando argumentos da outra classe:
2 pedro.Anda(praia.lugar)
```

```
Pedro está andando até Praia de Ipanema
```

Exemplo:

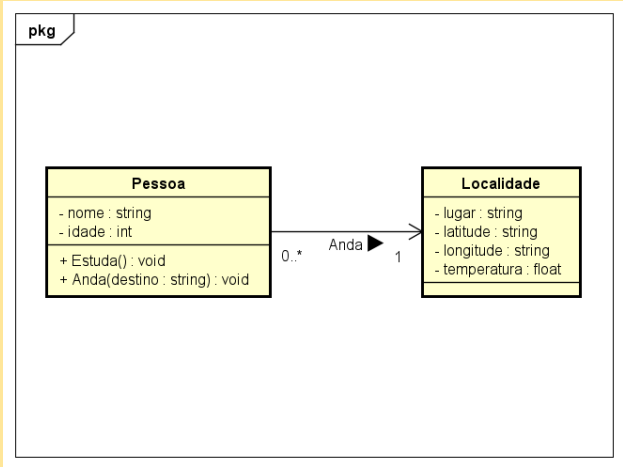
Teste de uma classe chamando a outra

```
1 # Testando a instância chamando o método e passando argumentos da outra classe:
2 pedro.Anda(praia.lugar)
```

Nesta aula

- Conceito.
- Representação.
- Desenvolvimento.
- **Resumo.**

Resumo

Conceito	Representação em UML	Desenvolvimento em Python
Classe: é uma representação de um item do mundo. Uma classe possui: **Atributo**: usado para armazenar os dados dos objetos de uma classe; **Método**: operações que a instância pode receber ou executar. **Objeto** é a instância de uma classe.		Revisão [ ] ``` # Criação da classe Pessoa class Pessoa: '''Classe Pessoa''' def __init__(self, nome, idade): self.nome = nome self.idade = idade def Estuda(self): print(self.nome + " está estudando...") def Anda(self, destino): print(self.nome + " está andando até " + destino) ``` [ ] ``` # Criação da classe Localidade: class Localidade: '''Classe Localidade''' def __init__(self, lugar, latitude, longitude, temperatura): self.lugar = lugar self.latitude = latitude self.longitude = longitude self.temperatura = temperatura ``` ``` # Definição de uma instância: pedro = Pessoa('Pedro', 18) # Definição de uma instância: praia = Localidade('Praia de Ipanema', '-22.9836210655', '-43.2027158558', 42.8) ``` [ ] ``` # Testando a instância chamando o método e passando argumentos da outra classe: pedro.Anda(praia.lugar) ``` Pedro está andando até Praia de Ipanema

Referências

- GUEDES, G. *UML 2 – Guia Prático* - 2ª Edição. São Paulo: Editora Novatec, 2007.
- PRESSMAN, R.S. MAXIM, B.R. *Software Engineering:* A Practitioner's Approach. 9th Edition. McGraw-Hill. 2020.
- BEZERRA, E. *Princípios de Análise e Projeto de Sistemas com UML*. LTC. 2014.