Programação Orientada a Objetos com Ruby

Tenille Martins

DIO Tech Education

 **bytemartins**

 **Tenille10**

 **Tenille Martins**

Objetivo Geral

Este curso foi planejado para programadores com conhecimento básico de Ruby; ao final deste curso, o DEV conseguirá codificar orientado a objeto em Ruby.

Pré-requisitos

- Um computador com acesso a internet;
- Muita vontade de aprender;
- Módulo I – Introdução ao Ruby
- Módulo II – Métodos e Gems

Percurso

Etapa 1	POO
Etapa 2	Classe
Etapa 3	Objetos

Percurso

Etapa 4	Exemplos na prática
Etapa 5	Require
Etapa 6	Escopo das variáveis

Percurso

Etapa 7 Atributos

Etapa 8 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

POO

// POO

Percurso

Etapa 1	**POO**
Etapa 2	Classe
Etapa 3	Objetos

Percurso

Etapa 4	Exemplos na prática
Etapa 5	Require
Etapa 6	Escopo das variáveis

Percurso

Etapa 7 Atributos

Etapa 8 Construtores

O que é POO?

- Programação Orientada a Objetos

Objetivo

- Aproximar o mundo digital do mundo real

Antes da POO

- Programação de Baixo nível
- Programação linear
- Programação estruturada
- Programação Modular
- POO

Como surgiu

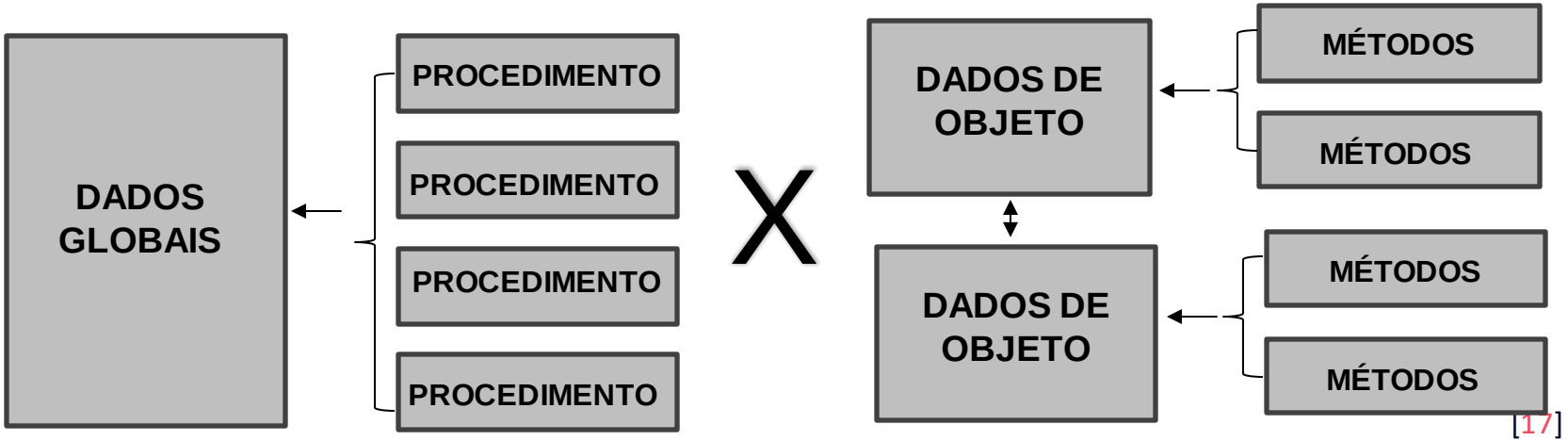
- Idealizada por Alan Kay
- Matemático
- Biológo
- 1970

Postulado

- “**o computador ideal deveria funcionar como um organismo vivo**, isso é, cada célula se relaciona com outras a fim de alcançar um objetivo, mas cada uma funciona de forma autônoma. As células poderiam também reagrupar-se para resolver outro problema, ou desempenhar outras funções”.

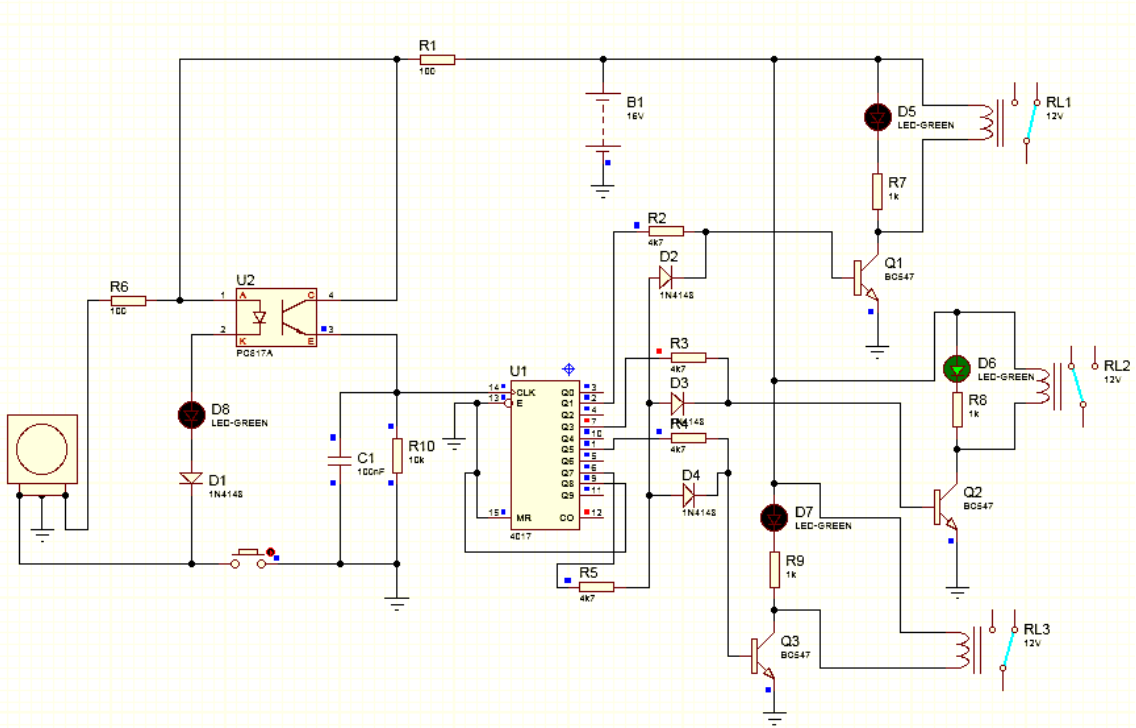
Antes POO x Depois POO

- “**o computador ideal deveria funcionar como um organismo vivo**, isso é, cada célula se relaciona com outras a fim de alcançar um objetivo, mas cada uma funciona de forma autônoma. As células poderiam também reagrupar-se para resolver outro problema, ou desempenhar outras funções”.



Exemplo Mundo Real

Linguagem não OO



Linguagem OO

Vantagens

1. Confiável: Como as partes são isoladas temos um software mais seguro. Esse isolamento permite que alteramos apenas uma das partes sem alterar as outras
2. Oportuno: Como dividimos em partes, elas podem ser desenvolvidas em paralelo

Vantagens

3. Manutenível: De fácil manutenção ou atualização. Ao modificar uma única parte você pode beneficiar todas as partes que usam o mesmo objeto

4. Extensível: O software deve sempre crescer, para que permaneça útil.

Vantagens

5. Reutilizável: usar um objeto de um sistema antigo para desenvolver um novo sistema.
** reutilizar o mesmo objeto em diferentes partes do projeto

6. Natural: Fácil de entender. A principal preocupação é a funcionalidade e não os detalhes

Objetos

Objetos são abstrações do mundo real ou entidades do sistema que se auto gerenciam.
Objetos são independentes e encapsulam representações de informação e estado.
A funcionalidade do sistema é expressa em termos de serviços dos objetos.
Objetos se comunicam por passagem de mensagem.

Domínios

- É uma estrutura de classificação de elementos que se relacionam

Em sistemas OO suas classes estão em um dos seguintes domínios:

– Domínio de aplicação
– Domínio de negócio
– Domínio de arquitetura
– Domínio de base

Domínio de Base

- O domínio de base descreve classes fundamentais, estruturais e semânticas

Domínio de Arquitetura

- O domínio de arquitetura fornece abstrações para a arquitetura de hardware ou software utilizada

Domínio de Negócio

- O domínio de negócio descreve classes inerentes a uma determinada área do conhecimento

Domínio de Aplicação

- O domínio de aplicação descreve classes “cola”, que servem para fazer as classes dos demais domínios funcionarem em um sistema

Resumo da Camada Domínio

Contém toda informação sobre o domínio e é considerado o coração do projeto. Aqui é mapeado os objetos e comportamentos do mundo real para o software.

4 Pilares da POO

- Abstração.
- Encapsulamento.
- Herança.
- Polimorfismo.

Abstração

Como estamos lidando com uma representação de um objeto real (POO), temos que imaginar o que esse objeto irá realizar dentro de nosso código. São três pontos que devemos levar em consideração nessa abstração.

1. Identidade do Objeto; (Ex: Controle)
2. Propriedades do Objeto; (Ex: Tamanho)
3. Métodos do Objeto; (Ex: Ligar)

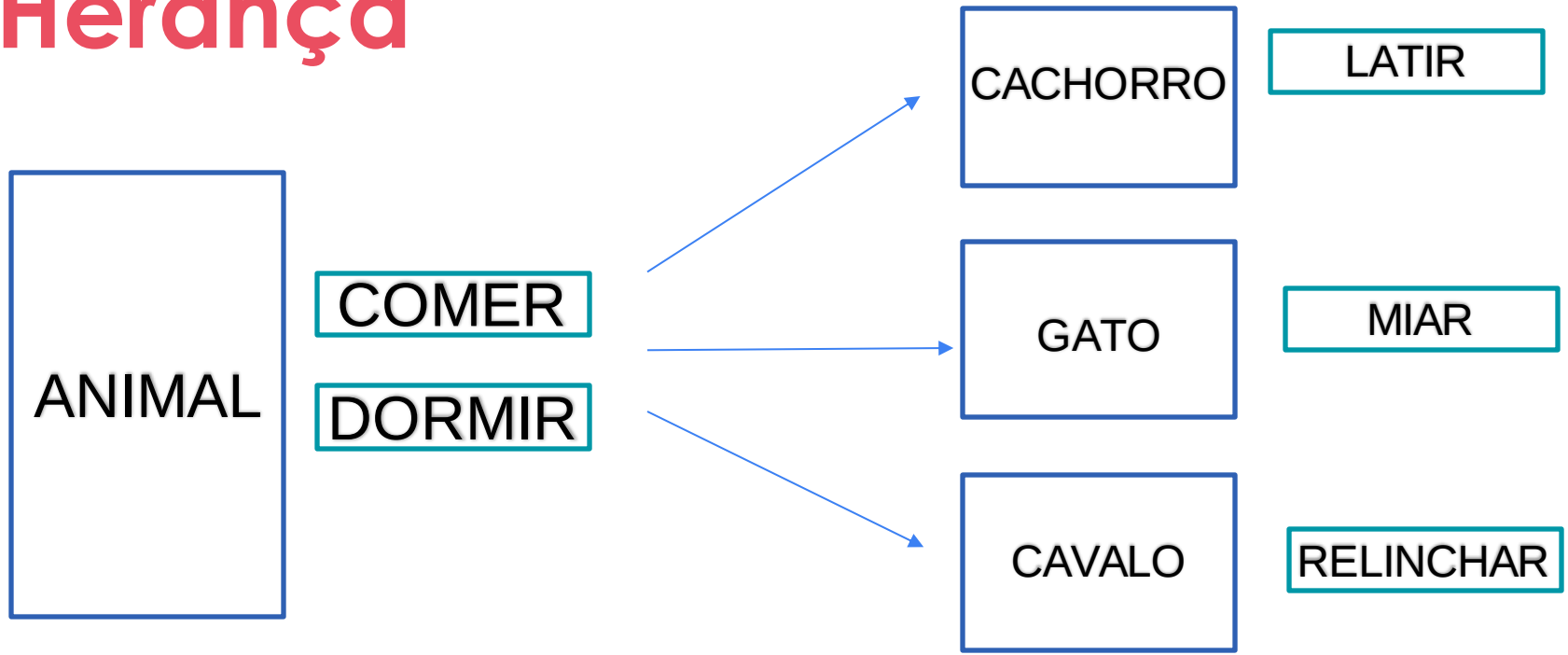
Encapsulamento

É um elemento que adiciona segurança à aplicação em uma POO pelo fato de esconder as propriedades, criando uma espécie de blindagem.
Resumindo: Evitar que eles tenham acesso indevido.
Ex: Para ligar a TV usando nosso controle você não precisa ter acesso a parte interna do controle, garantindo assim que você não vai alterar alguma função que você não domina e o controle parará de funcionar.

Herança

reuso de código é uma das grandes vantagens da programação orientada a objetos.
O objeto abaixo na hierarquia irá herdar características de todos os objetos acima dele, seus “ancestrais”. A herança a partir das características do objeto mais acima é considerada herança direta, enquanto as demais são consideradas heranças indiretas.

Herança


ANIMAL
COMER
DORMIR
CACHORRO
LATIR
GATO
MIAR
CAVALO
RELINCHAR

Polimorfismo

De forma genérica, **polimorfismo** significa "várias formas".
Ex do dia-a-dia: A variação canhoto/destro é um polimorfismo;
Como vimos em herança, os objetos filhos herdam as características e ações de seus “ancestrais”. Entretanto, em alguns casos, é necessário que as ações para um mesmo método seja diferente.

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Classe

// POO

Percurso

~~Etapa 1~~ ~~POO~~

Etapa 2 **Classe**

Etapa 3 Objetos

Percurso

Etapa 4	Exemplos na prática
Etapa 5	Require
Etapa 6	Escopo das variáveis

Percurso

Etapa 7 Atributos

Etapa 8 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Objetos

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

Etapa 3 Objetos

Percurso

Etapa 4	Exemplos na prática
Etapa 5	Require
Etapa 6	Escopo das variáveis

Percurso

Etapa 7 Atributos

Etapa 8 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Prática: abstração e encapsulamento

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

~~Etapa 3~~ ~~Objetos~~

Percurso

Etapa 4 **Prática: Abstração e encapsulamento**

Etapa 5 Prática: Herança

Etapa 6 Prática: Polimorfismo

Percurso

Etapa 7	Require
Etapa 8	Escopo das variáveis
Etapa 9	Atributos

Percurso

Etapa 10 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Prática: Herança

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

~~Etapa 3~~ ~~Objetos~~

Percurso

~~Etapa 4~~	~~**Prática: Abstração e encapsulamento**~~
Etapa 5	**Prática: Herança**
Etapa 6	Prática: Polimorfismo

Percurso

Etapa 7	Require
Etapa 8	Escopo das variáveis
Etapa 9	Atributos

Percurso

Etapa 10 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Prática: Polimorfismo

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

~~Etapa 3~~ ~~Objetos~~

Percurso

~~Etapa 4~~ ~~Prática: Abstração e encapsulamento~~

~~Etapa 5~~ ~~Prática: Herança~~

Etapa 6 Prática: Polimorfismo

Percurso

Etapa 7	Require
Etapa 8	Escopo das variáveis
Etapa 9	Atributos

Percurso

Etapa 10 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Require

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

~~Etapa 3~~ ~~Objetos~~

Percurso

~~Etapa 4~~ ~~Prática: Abstração e encapsulamento~~

~~Etapa 5~~ ~~Prática: Herança~~

~~Etapa 6~~ ~~Prática: Polimorfismo~~

Percurso

Etapa 7	**Require**
Etapa 8	Escopo das variáveis
Etapa 9	Atributos

Percurso

Etapa 10 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Escopo de variáveis

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

~~Etapa 3~~ ~~Objetos~~

Percurso

~~Etapa 4~~ ~~Prática: Abstração e encapsulamento~~

~~Etapa 5~~ ~~Prática: Herança~~

~~Etapa 6~~ ~~Prática: Polimorfismo~~

Percurso

~~Etapa 7~~	~~**Require**~~
Etapa 8	**Escopo das variáveis**
Etapa 9	Atributos

Percurso

Etapa 10 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Atributos

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

~~Etapa 3~~ ~~Objetos~~

Percurso

~~Etapa 4~~ ~~Prática: Abstração e encapsulamento~~

~~Etapa 5~~ ~~Prática: Herança~~

~~Etapa 6~~ ~~Prática: Polimorfismo~~

Percurso

~~Etapa 7~~ ~~**Require**~~

~~Etapa 8~~ ~~**Escopo das variáveis**~~

Etapa 9 Atributos

Percurso

Etapa 10 Construtores

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Construtores

// POO

Percurso

~~Etapa 1~~ ~~POO~~

~~Etapa 2~~ ~~Classe~~

~~Etapa 3~~ ~~Objetos~~

Percurso

~~Etapa 4~~ ~~Prática: Abstração e encapsulamento~~

~~Etapa 5~~ ~~Prática: Herança~~

~~Etapa 6~~ ~~Prática: Polimorfismo~~

Percurso

~~Etapa 7~~ ~~Require~~

~~Etapa 8~~ ~~Escopo das variáveis~~

~~Etapa 9~~ ~~Atributos~~

Percurso

Etapa 10 **Construtores**

Links Úteis

- **Repositório no GitHub:** https://github.com/Tenille10/
- **Documentação Oficial: https**: www.ruby-lang.org
- **Referências:** Ruby: Aprenda a programar na linguagem mais divertida.

Para saber mais

- **Site na Web:** www.ruby-lang.org

- **Artigo:** https://pt.wikipedia.org/wiki/Programa%C3%A7%C3%A3o_orientada_a_objetos

- **Livro:** Orientação a Objetos: Aprenda seus conceitos e suas aplicabilidades de forma efetiva.

- .

Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)