UNIP UNIVERSIDADE PAULISTA

- Professor: Yure de Queiroz Lima
- E-mail: yureql@hotmail.com

Objetivo

- Aproximar o mundo digital ao mundo real!

Antigamente

- Linguagem de baixo nível
  1. Binário
  2. Decimal
  3. .....

UNIP
UNIVERSIDADE PAULISTA

Programação Orientada a Objetos

Alan Kay

Ele lançou o postulado de que o computador ideal deveria funcionar como um organismo vivo, isto é, cada "célula" comportar-se-ia relacionando-se com outras células a fim de alcançar um objetivo, entretanto, funcionando de forma autônoma. As células poderiam também reagrupar-se para resolver outros problemas ou desempenhar outras funções, trocando mensagens "químicas" entre elas.

Histórico

- A **orientação a objetos** tem sua origem nos anos 60 na Noruega.
- No Centro Norueguês de Computação. Através da linguagem Simula 67, foram introduzidos os conceitos de classe e herança.
- A orientação a objetos foi mais bem conceituada no laboratório da Xerox, em Palo Alto, sendo refinada numa seqüência de protótipos da linguagem Smalltalk.

Histórico

- O líder desse projeto foi Alan Curtis Kay, considerado um dos criadores do termo “programação orientada a objetos”.
- Então, Alan Kay pensou em como construir um sistema de software a partir de agentes autônomos que interagissem entre si, estabelecendo os seguintes princípios da orientação a objetos:

Histórico

- Qualquer coisa é um objeto.
- Objetos realizam tarefas através da requisição de serviços.
- Cada objeto pertence a uma determinada classe.
- Uma classe agrupa objetos similares.
- Um classe possui comportamentos associados ao objeto.
- Classes são organizadas em hierarquias.

Introdução

- O conceito de Classe foi introduzido na década de 60 através da LP Simula 67.
- A primeira LP OO pura foi Smalltalk desenvolvida durante a década de 70.
- Outros exemplo de LPs OO são Eiffel, Java, Object Pascal e C#.
- LPs como C++, Ada e Perl suportam o conceito de classe mas não são consideradas LPs OO em um sentido mais rigoroso.

Vantagens

- Decorar: COMERNada : 6 Vantagens
- **Confiável:** isolamento entre as partes.
- **Oportuno:** Desenvolver em paralelo, dividindo tudo em partes.
- **Manutenível:** Manutenção mais fácil.
- **Extensível:** Não é estático. Deve crescer para permanecer útil.
- **Reutilizável:** Podemos utilizar objetos de um sistema em outro criado futuramente.
- **Natural:** Mais fácil de entender. Você se preocupa mais na funcionalidade do que nos detalhes de implementação.

Introdução

- Com o desenvolvimento de aplicações de software cada vez mais complexas, cresceram as demandas por metodologias que pudessem abstrair e modularizar as estruturas básicas de programas.
- A programação OO suporta a abstração e modularização de dados (as classes), e promove a reutilização de software por meio do mecanismo de herança.

Introdução

- A programação OO busca modelar aplicações seguindo uma estrutura semelhante àquela encontrada no mundo real.
- O mundo é composto por objetos que apresentam estados e comportamentos.
- Um gato por exemplo tem um nome, uma cor e uma raça (os estados) e ele come, mia e caça ratos (os comportamentos).

Objeto

- O que é um objeto....

Objeto

- Coisa material ou abstrata que pode ser percebida pelos sentidos e descrita por meio das suas características, comportamentos e estado atual.

Modelo OO

- Uma aplicação OO é composta por diferentes objetos e uma seqüência de ações (interações).
- Uma ação se inicia através do envio de uma mensagem para um agente (um objeto) que será responsável por tratar essa ação.
- A mensagem carrega uma requisição além de toda a informação necessária (argumentos) para que a ação seja executada.
- Se o agente receptor da mensagem à aceita, ele tem a responsabilidade de executar um método para cumprir a requisição.

Classe sempre tem de responder 3 perguntas

- **Coisas que eu tenho:** Modelo, cor, ponta, carga, tampada.
- **Coisas que eu faço:** Escrever, pintar, tampar, destampar.
- **Como eu estou agora:** esta com 30 % de carga

Classe sempre tem de responder 3 perguntas

- **Atributos:** Modelo, cor, ponta, carga, tampada.
- **Métodos:** Escrever, pintar, tampar, destampar.
- **Estado:** esta com 30 % de carga

O que devemos Saber!
Conceitos essenciais:

- **Classe:** representa um conjunto de objetos com características afins. Uma classe define o comportamento dos objetos através de seus métodos, e quais estados ele é capaz de manter através de seus atributos.
- **Exemplo de classe:** Os seres humanos
- **Subclasse:** É uma nova classe que herda características de sua(s) classe(s) ancestral(is)

O que devemos Saber!
Conceitos essenciais:

- **Objeto / Instância de uma classe:** Um objeto é capaz de armazenar estados através de seus atributos e reagir a mensagens enviadas a ele, assim como se relacionar e enviar mensagens a outros objetos
- **Exemplo de objetos da classe Humanos:** João, José, Maria

Instanciar

- **Geramos um objeto através de uma classe. Instanciamos uma classe em forma de um objeto.**
- c1 = nova Caneta
- c1.cor = “Preta”
- c1.ponta = 0.5
- c1.tampada = falso
- c1.escrever()
- c2 = nova Caneta

Definição...

- **Classe:** Define os atributos e métodos comuns que serão compartilhados por um objeto.
- **Objeto:** É a instância de uma classe.

O que devemos Saber!
Conceitos essenciais:

- **Atributo:** são características de um objeto. Basicamente a estrutura de dados que vai representar a classe.
- **Exemplos:** Funcionário: nome, endereço, telefone, CPF,...; Livro: autor, editora, ano. Os atributos possuem valores. Por exemplo, o atributo cor pode conter o valor azul. O conjunto de valores dos atributos de um determinado objeto é chamado de estado

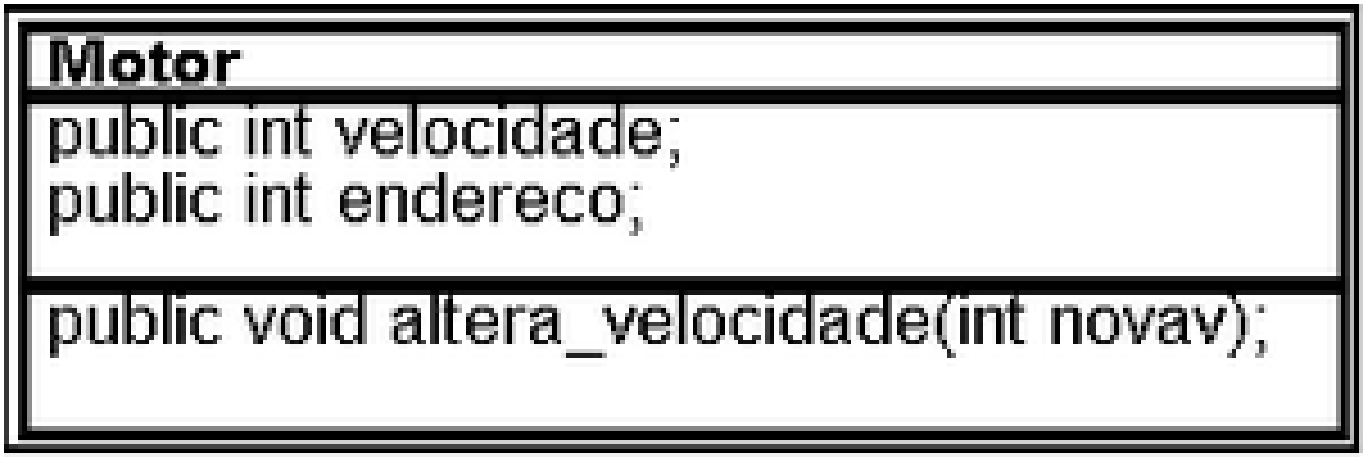
O que devemos Saber!
Conceitos essenciais:

- **Método:** definem as habilidades dos objetos. Toto é uma instância da classe Cachorro, portanto tem habilidade para latir, implementada através do método deUmLatido. Um método em uma classe é apenas uma definição. A ação só ocorre quando o método é invocado através do objecto, no caso Toto.

O que devemos Saber!

Conceitos essenciais:

- **Um diagrama simplificado da classe motor com os atributos e métodos:**



O que devemos Saber!
Conceitos essenciais:

- Dentro do programa, a utilização de um método deve afetar apenas um objeto em particular; Todos os cachorros podem latir, mas você quer que apenas Toto dê o latido. Normalmente, uma classe possui diversos métodos, que no caso da classe Cachorro poderiam ser somente, coma e morda.

O que devemos Saber!
Conceitos essenciais:

- **Mensagem:** é uma chamada a um objeto para invocar um de seus métodos, ativando um comportamento descrito por sua classe. Também pode ser direcionada diretamente a uma classe (através de uma invocação a um método estático)