Programação Orientada a Objetos

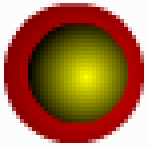
Encapsulamento e Sobrecarga de Métodos na Classe

André Santanchè e Oscar Rojas
Institute of Computing – UNICAMP

Março 2015

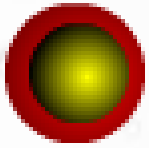
Picture by http://www.flickr.com/photos/crouchy69, 2010

Encapsulamento



Princípios do Paradigma
Encapsulamento

- "Objetos do mundo real encapsulam em si os próprios atributos, quer sejam descritivos, partes componentes ou funções.” (Meyer, 1997)
  - Objetos e mensagens
    - “Na programação orientada a objetos, os objetos comunicam-se entre si através de mensagens. A única coisa que um objeto conhece sobre outro objeto é a sua interface de comunicação. Os dados e a lógica de cada objeto são mantidos escondidos dos outros objetos. Em outras palavras, a interface encapsula o código e os dados do objeto”. (IBM)



Encapsulamento
Interface x Implementação

- **Interface**: descreve como as partes do objeto se relacionam com o exterior.

- **Implementação**: dados e código que implementam o comportamento dos objetos da classe; esta parte não é visível externamente.

Encapsulamento
Níveis de Acesso

- **Privada**: não visível a nenhuma classe externa; visível apenas dentro da classe.
- **Pública**: completamente visível internamente e para outras classes e elementos externos.
- **Protegida**: não visível a classes e elementos externos; visível apenas dentro da classe e para seus herdeiros.

Encapsulamento
Atributos Privados e Métodos Públicos

- Na POO uma classe possui, em geral, os **atributos privados** e os **métodos podem ser públicos**, tornando o objeto como uma caixa preta onde só aparece o suficiente para que o programa possa utilizá-lo.

Java
Níveis de Acesso

- **Privada (private)**: visível exclusivamente dentro da classe.
- **Pública (public)**: completamente visível dentro e fora da classe.
- **Protegida (protected)**: visível apenas dentro da classe, pelos seus herdeiros e pelas classes no mesmo pacote.
- **Pacote (*padrão*)**: visível apenas dentro da classe e pelas classes que estão no mesmo pacote.

Atributo final

- Atributo que recebe o prefixo final
- Só pode receber o valor uma vez na declaração ou no construtor
- Depois que recebe o valor não pode ser modificado
- Usualmente associado ao static para a criação de uma constante

Sobrecarga de Métodos

Assinatura de um Método

- Envolve
  - nome do método
  - número de parâmetros
  - tipos dos parâmetros
- Métodos que não tenham o mesmo nome, mesmo número de parâmetros com os mesmos tipos (na ordem) não têm a mesma assinatura

Sobrecarga de Métodos

- Sobrecarga de método: técnica que envolve criar vários métodos com o mesmo nome e implementações diferentes.
- Tipos:
  - sobrecarga na mesma classe
    - assinaturas têm que ser diferentes
    - identificados pela assinatura
  - sobrecarga em classes herdeiras
    - assinaturas podem ser iguais ou diferentes
    - tratado na aula de herança

JavaScript

- Não permite mais de um método com o mesmo nome
  - Se for feito, o segundo método substitui o primeiro
- Pode ser simulado:
  - pela verificação de parâmetros “undefined”
  - uso de vetor de parâmetros

Referências Bibliográficas

- Almeida, Charles Ornelas , Guerra, Israel; Ziviani, Nivio (2010) **Projeto de Algoritmos** (transparências aula).
- Bloom, Paul (2007) **Introduction to Psychology** – transcrição das aulas (aula 17). Yale University.
- Ferreira, Aurélio B. H. (1989) **Minidicionário da Língua Portuguesa**. Rio de Janeiro, Editora Nova Fronteira.
- Houaiss, Instituto Antônio. **Dicionário Houaiss da língua portuguesa** (2006) Editora Objetiva, Março.
- IBM - International Business Machines Corporation. **IBM Smalltalk Tutorial** [Online] http://www.wi2.uni-erlangen.de/sw/smalltalk/
- Liskov, Barbara; Zilles, Stephen. **Programming with abstract data types** (1974) ACM SIGPLAN Notices, 9 (4) p. 50.

Referências Bibliográficas

- Meyer, Bertrand (1997) **Object-Oriented Software Construction - Second Edition**. USA, Prentice-Hall, Inc.
- Miller, Robert (2004) **6.831 User Interface Design and Implementation (lecture notes)**. MIT OpenCourseware.
- Rocha, Heloisa Vieira da, Baranauskas, Maria Cecilia Calani (2003) **Design e Avaliação de Interfaces Humano-Computador.** NIED/UNICAMP.
- Santos, L. R., & Hood, B. M. (2009). **Object representation as a central issue in cognitive science**. The Origins of Object Knowledge: The Yale Symposium on the Origins of Object & Number Representation. Oxford: Oxford University Press.
- Shaw, M. **Abstraction Techniques in Modern Programming Languages** (1984) IEEE Software, 1, 4, 10-26.

Referências Bibliográficas

- Tenenbaum, Aaron M.; Langsam, Yedidyah; Augenstein, Moshe J. **Data Structures Using C** (1990) Prentice Hall, Upper Saddle River, NJ.

André Santanchè

http://www.ic.unicamp.br/~santanche

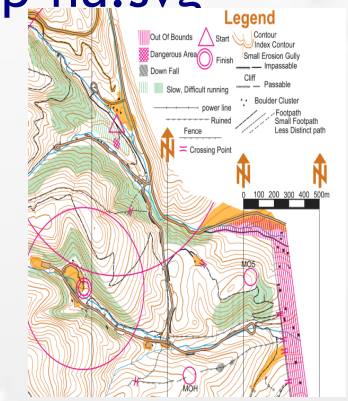
License

- These slides are shared under a Creative Commons License. Under the following conditions: Attribution, Noncommercial and Share Alike.
- See further details about this Creative Commons license at: http://creativecommons.org/licenses/by-nc-sa/3.0/

Imagens Externas

- Havang(nl) [http://commons.wikimedia.org/wiki/User:Havang%2
  url (ver licença específica):
  http://commons.wikimedia.org/wiki/File:Bomenpark_Meijhorst,_Nijmegen_%28

- Eric Gaba [http://commons.wikimedia.org/wiki/User:Sting]
  url (ver licença específica):
  http://commons.wikimedia.org/wiki/File:Easter_Island_map-hu.svg



- Kharker [http://en.wikipedia.org/wiki/User:Kharker]
  url (ver licença específica):
  http://commons.wikimedia.org/wiki/File:Ardf_map.png