Objetivos Atingidos

Aline Marcelo Garlet Millani

15 de junho de 2013

1

- Definição e uso de classes
  1. Particle (Particle.ml)
  2. Electric (Particle.ml)
  3. Body (Body.ml)
- Encapsulamento e proteção dos atributos
  1. Todos os atributos são privados
  2. métodos como move e force mostram o encapsulamento
- Organização do código em espaços de nome diferenciados
  1. Para se usar os módulos, é necessário ou prefixar as funções com o nome deles ou usar open
- Mecanismo de herança:
  - especificação de 3 níveis de hierarquia
    1. Body (Body.ml)
    2. Particle (Particle.ml)
    3. Electric (Electric.ml)
  - especificação de uma classe abstrata
    1. Body (Body.ml)
  - polimorfismo por inclusão
- Polimorfismo paramétrico
  - especificação de algoritmo utilizando o recurso
    1. buildQuadtree (Quadtree.ml)
  - especificação de estrutura de dados genérica
    1. tipo quadtree (Quadtree.ml)
- Polimorfismo por sobrecarga

1. Ocaml não suporta polimorfismo por sobrecarga

- Especificação e uso de funções como elementos de primeira ordem
  1. readParticles (Loader.ml)
- Especificação e uso de funções de ordem maior
  1. loadConfig (Loader.ml)
  2. parseChannel (Loader.ml)
- Uso de lista para manipulação de estruturas em funções de ordem maior (as funções devem ser puras)
- Uso de funções lambda
  1. readElectricParticle (Loader.ml)
  2. buildQuadtree (Quadtree.ml)
- Currying
  1. display (main.ml)
  2. buildQuadtree (main.ml)
- Pattern matching
  1. drawDots (main.ml)
  2. moveDots (Physics.ml)
- Recursão como mecanismo de iteração
  1. drawDots (main.ml)
  2. moveDots (Physics.ml)
- Delegates