DigitalHouse>
Coding School

Certified Tech Developer
The Ultimate Degree

Back End I

Exercício Mesas de Trabalho: Flyweight pattern

Objetivo

Realizar o diagrama **UML** e **programar em Java**, implementando o **Flyweight pattern** de acordo com o seguinte enunciado.
Considere que:

- Exercicio individual
- Nível de complexidade intermediário: 🔥🔥

Enunciado

Em uma empresa, eles precisam plantar árvores para poder ver quanto espaço ocupariam. O programa que existe atualmente tem um alto consumo de recursos. Você precisa plantar 1.000.000 de árvores. O processo de plantação das árvores considera que cada **árvore** possui uma Altura, Largura, cor e tipo de árvore. Os tipos de árvores que existem são:

- Ornamentais
    - Altura: 200.
    - Largura: 400.
    - Cor: verde.
- Frutíferas
    - Altura: 500.
    - Largura: 300.
    - Cor: vermelho.
- Florífera
    - Altura: 100.
    - Largura: 200.
    - Cor: azul.

A **floresta** é um conjunto de árvores e permitirá o plantio de árvores. **ArvoreFactory** será o local onde os diferentes tipos de árvores serão armazenados. Isso permitirá, antes de criar o objeto, validar se já existe um idêntico ao que está sendo solicitado. Em caso afirmativo, retorna o objeto existente; se não existir, ele cria o novo objeto e o armazena em cache para ser reutilizado posteriormente.

Você precisa de um sistema que mostre 1.000.000 de árvores, metade vermelhas e metade verdes e, em seguida, imprima na tela as árvores e quanta memória está sendo utilizada.

Usando esta declaração, você pode ver a memória usada:

Runtime runtime = Runtime.getRuntime();

System.out.println("Memória utilizada: " + (runtime.totalMemory() -

runtime.freeMemory()) / (1024 * 1024));

As classes necessárias devem ser desenvolvidas para atingir o processo de preparação explicado.

Bons estudos!