Universidade Federal de Mato Grosso do Sul
Faculdade de Computação

Linguagem de Programação Orientada a Objetos 2025

Paulo A. Pagliosa

TRABALHO 2

11/07/2025

Imagine um motor de jogo hipotético no qual um **jogo** é definido por uma **lista de cenas**. Uma **cena**, S, é caracterizada por um nome, dado por uma cadeia de caracteres, e uma **lista de objetos de jogo**, R_S, chamada *raiz*. Qualquer **objeto de jogo**, O, também pode conter sua própria lista de objetos de jogo, H_O, os *filhos* de O. O objeto de jogo O é *pai* dos objetos de jogo contidos em H_O. Um objeto de jogo tem um nome, dado por um cadeia de caracteres, e conhece seu objeto de jogo pai e também a cena em cuja hierarquia está contido (um objeto de jogo "atua" em somente uma cena). Os objetos de jogo pertencentes a R_S, ou seja, a raiz de sua cena, não possuem pai.

Um objeto de jogo, O, contém ainda uma **lista de componentes**, C_O. Um **componente** encapsula certas propriedades e funcionalidades do objeto de jogo. Todo componente conhece o objeto de jogo do qual faz parte. Todo objeto de jogo (mesmo um objeto de jogo vazio, isto é, sem filhos) contém um componente do tipo **transformação** (em um motor de jogo real, como o Unity,[1] por exemplo, este componente especifica a posição, orientação e tamanho do objeto de jogo). Um objeto de jogo pode ter outros tipos de componentes, mas C_O não pode conter mais de um componente de um mesmo tipo. Neste trabalho, componentes podem ser ainda dos tipo **malha**, **luz** ou **câmera**, conforme especificado no diagrama de classes UML da Figura 1.

Tarefas

O trabalho consiste na implementação em Java das classes de objetos do diagrama da Figura 1, bem como em uma aplicação de teste das classes. As atividades de programação são:

A1 Criação e operações de um jogo. Quando um novo jogo, G, é criado, este conterá uma nova cena chamada `Main Scene`. As operações sobre um jogo envolvem:

- Adição de uma nova cena à lista de cenas de G;
- Escrita de uma cena da lista de cenas de G em arquivo, explicado em sala;
- Leitura de uma cena de arquivo e adição à lista de cenas de G;
- Remoção de uma cena da lista de cenas de G, dado o nome da cena;
- Remoção de todas as cenas da lista de cenas de G;
- Obtenção de uma cena da lista de cenas de G, dado o nome da cena;
- Atravessamento da lista de cenas de G, a fim de se executar uma função qualquer para cada cena; e
- Exibição do jogo. Exibir o jogo significa exibir cada uma das cenas da lista de cenas de G.

[1] `https://unity.com/pt/games`.

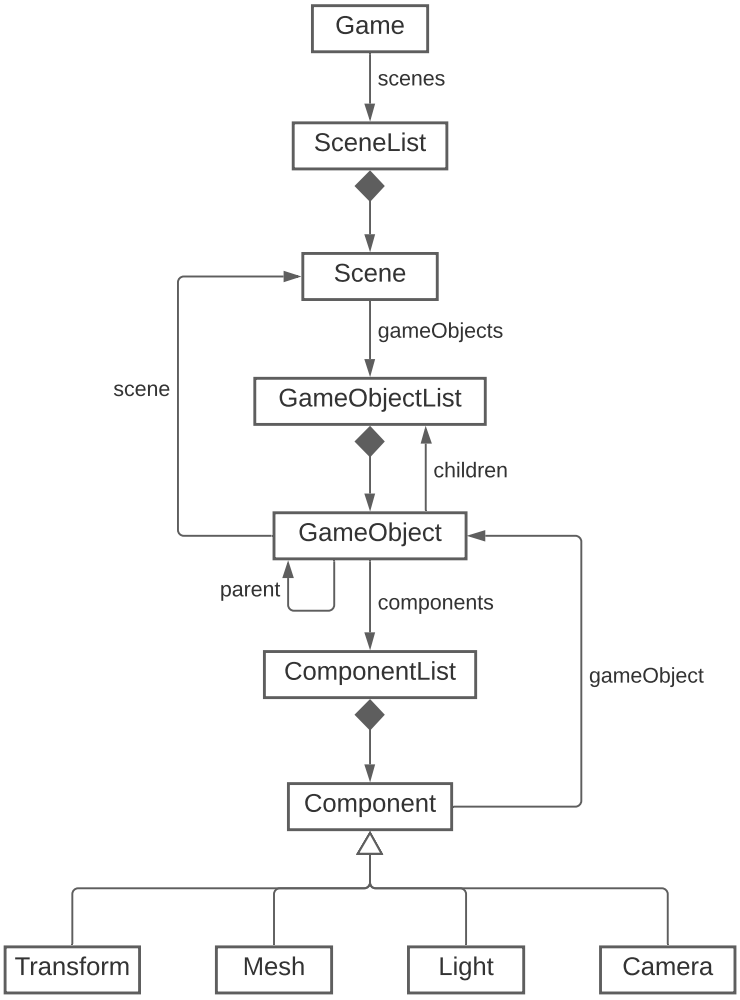

Figura 1: Diagrama de classes UML do modelo proposto.

A2 Criação e operações de uma cena. Quando uma nova cena, S, é criada, esta conterá três novos objetos de jogo chamados (1) `Cube`, (2) `Light`, e (3) `Main Camera`. Além de um componente do tipo `Transform`, obrigatório para todo objeto de jogo, o objeto de jogo (1) conterá um componente do tipo `Mesh`, o objeto de jogo (2) conterá um componente do tipo `Light` e o objeto de jogo (3) conterá um componente do tipo `Camera`. As operações sobre uma cena envolvem:

- Obtenção e alteração do nome da cena;
- Adição de um novo objeto de jogo à lista de objetos de jogo de S;
- Remoção de um objeto de jogo da lista de objetos de jogo de S, dado o nome do objeto;
- Remoção de todas os objetos de jogo da lista de objetos de jogo de S;
- Obtenção de um objeto de jogo da lista de objetos de jogo de S, dado o nome do objeto;
- Atravessamento da lista de objetos de jogo de S, a fim de se executar uma função qualquer para cada objeto; e
- Exibição da cena. Exibir a cena S significa imprimir, na saída padrão, o nome de S, e em seguida exibir cada um dos objetos de jogo da raiz de S.

A3 Criação e operações de um objeto de jogo. Quando um novo objeto de jogo, O, é criado, este é vazio e conterá um novo componente do tipo `Transform`, o qual não poderá ser removido da lista de componentes de O (enquanto o objeto de jogo existir). As operações sobre um objeto de jogo envolvem:

- Obtenção e alteração do nome do objeto de jogo;
- Adição de um novo objeto de jogo à lista de objetos de jogo filhos de O;
- Remoção de um objeto de jogo da lista de objetos de jogo de filhos de O, dado o nome do objeto;
- Remoção de todas os objetos de jogo da lista de objetos de jogo filhos de O;
- Obtenção de um objeto de jogo da lista de objetos de jogo filhos de O, dado o nome do objeto;
- Atravessamento da lista de objetos de jogo filhos de O, a fim de se executar uma função qualquer para cada objeto;
- Adição de um novo componente à lista de componentes de O, desde que já não exista um daquele tipo na lista;
- Remoção de um componente da lista de componentes de O, dado o nome da classe do componente (exceto `Transform`);
- Remoção de todos os componentes da lista de componentes de O (exceto `Transform`);
- Obtenção de um componente da lista de componentes de O, dado o nome da classe do componente;
- Atravessamento da lista de componentes de O, a fim de se executar uma função qualquer para cada componente; e
- Exibição do objeto de jogo. Exibir o objeto de jogo O significa imprimir, na saída padrão, o nome de O, o nome do objeto de jogo pai de O, se houver, em seguida inspecionar cada um dos componentes da lista de componentes de O e, finalmente, exibir cada objeto de jogo da lista de objetos de jogo filhos do objeto de jogo O.

A4 Criação e operações de componentes de um objeto de jogo. As operações sobre um componente, C, envolvem:

- Obtenção do nome da classe de C; e
- Inspeção do componente. Inspecionar um componente C significa imprimir, na saída padrão, o nome da classe de C, o nome do objeto de jogo contendo C e os atributos de C.

Os atributos de `Transform` são posição (vetor 3D), rotação (vetor 3D) e escala (número real). Os atributos de `Mesh` são o nome da malha e o nome do material, ambos dados por cadeias de caracteres. Os atributos de `Light` são o tipo (pontual ou direcional) e a cor (RGB) da luz. Os atributos de `Camera` são o tipo de projeção (perspectiva ou paralela), o ângulo de vista (número real) e as distâncias dos planos de recorte de frente e de fundo (par de números reais).

A5 Aplicação de teste. Escreva o "motor" de jogos com métodos para criação, edição e exibição de um ou mais jogos. Comece com a criação e exibição do jogo "padrão", isto é, aquele com uma cena com os três objetos de jogo descritos em **A2**. Então,

edite o jogo (adicionando e removendo cenas, adicionando e removendo objetos de jogo em uma cena e em outros objetos de jogo, adicionando e removendo componentes de um objeto de jogo, etc.) e salve o jogo em arquivo. Em seguida, remova o jogo do motor e o carregue do arquivo. Imprima na saída padrão um "log" de todas as operações de edição efetuadas em um jogo, seguida da exibição do jogo. Alternativamente, escreva uma interface em modo texto com comandos para que o usuário possa criar, editar e exibir um jogo. De um modo ou outro, o importante é que todas as funcionalidades descritas nessa seção sejam testadas.

Além dessas, o trabalho inclui:

A6 Vídeo (de no máximo 12 minutos) em que o(s) autor(es) aparece(m) 1) explicando as definições efetuadas no código fonte compartilhado em tela e 2) mostrando a compilação e execução do programa e seus resultados.

Entrega do trabalho

O trabalho deve ser entregue via AVA em arquivo único compactado chamado `t2.zip` contendo **somente** os arquivos de código fonte, sem quaisquer outros arquivos de projeto ou resultantes da compilação. Cada arquivo Java deve ter um comentário com identificação do(s) autor(es). Em adição, `t2.zip` deve conter um arquivo texto chamado `README.txt` com o nome do(s) autor(es), uma breve descrição de quais atividades foram parcial ou completamente concluídas ou não e um link para o vídeo (no YouTube ou Google Drive, por exemplo).

Não serão considerados e terão nota zero programas em que houver detecção de plágio de qualquer que seja a fonte, mesmo que parcialmente. Programas que não compilam ou não executam terão nota zero. O trabalho pode ser desenvolvido individualmente ou em duplas.

Lista de objetivos

A avaliação do trabalho levará em conta se:

- ☐ O arquivo `t2.zip` foi submetido com o conteúdo correto, incluindo `README.txt`.
- ☐ O vídeo em que o(s) autor(es) explica(m) o trabalho foi submetido.
- ☐ As atividades de programação **A1** até **A5** foram implementadas e a implementação está correta.
- ☐ Todos os métodos foram testados na aplicação de teste.