Software Development - 4g

Schroll, Ludvig
Barcij, Artur
Cadell, Samuel

March 4, 2022

Introduction

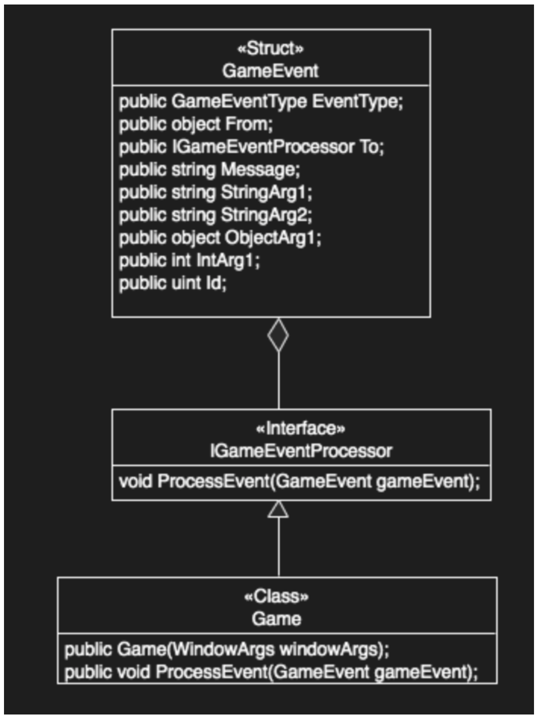
We were given the assignment to extend the implementation of the game enginge DIKUArcade and implement the game Galaga. This rapport will contain answers to four questions relating to the game.

Question 1: GameEventBus

Explain in brief terms the correlation between IGameEventProcessor, GameEvent and the delegation of key input events:

The type ”GameEvent” send messages to all objects that have implemented the interface ”IGameEventProcessor” which has the method ”ProcessEvent(GameEvent gameEvent)”. Through this method the objects implementing can subscribe to the messages that is sent from ”GameEvent”, which is important when responding to key-presses.

Here is a picture of our UML-diagram:


«Struct»
GameEvent
public GameEventType EventType;
public object From;
public IGameEventProcessor To;
public string Message;
public string StringArg1;
public string StringArg2;
public object ObjectArg1;
public int IntArg1;
public uint Id;
«Interface»
IGameEventProcessor
void ProcessEvent(GameEvent gameEvent);
«Class»
Game
public Game(WindowArgs windowArgs);
public void ProcessEvent(GameEvent gameEvent);

Question 2: Iterating shots

Explain what IEnumerable is:

When an object inherits the ”IEnumerable” interface then its elements can be iterated over. It defines the method ”GetEnumerator” which in turn returns the ”IEnumerator” interface.

Explain how the EntityContainer.Iterate() method works:

The ”Iterate” method inside the class ”EntityContainer” is used to iterate through the elements of the entity. It uses a simple for-loop to iterate through all the entities.

We see that EntityContainer implements IEnumerable, so when iterating through the container we could have used a foreach loop. Explain if using foreach would work differently than Iterate and if so, what the difference would be. Imagine a scenario where a difference could matter, and use that as a basis for your argumentation.

Foreach loop is iterates through whole array or list and if one needs to update each instance, e.g. in this assignment to update shot or animation, then one needs to implement manually this function/method.

Iterate method iterates through whole array/list/collection, while substantially updates each indexed value/instance. Thus the significant difference made these work in each their own way for different usage.

For instance it is better to implement iterate method, when multiple instances in one collection have to be updated after each iteration, e.g. positions, shapes, animations.

Question 3: Entities

Explain the differences between DynamicEntity and StationaryEntity, and how to convert between these types.

DynamicEntity is made for shapes, which can indulge in morphism, thus changing direction information based on physics library. This libary also includes void function ChangeDirection(), which takes direciton vector as argument and changes direction of it. In this assingment it was used for shot instances, which had to change direction to be shot upwards against enemies.

StationaryEntity does not change values if implemented, thus being stationary or static, since static objects does not contain direction information.

To convert from stationary to Dynamic one has to implement the direction information as direction vector.