# OOP – Environment System

The goal of this lab is to study the **Environment System** and extend its functionality. You are **NOT allowed to directly edit** any of the contents of the **Core** and **Interface** namespaces (only the **Generator** namespace).

## Overview

You are given a small **OOP project** that models an **environment** and simulates **different object behavior**.

Your first task is to **study the given project** in depth – go through the **classes**/**interfaces** and try to **get a good idea** of how the environment system works.

Here's a quick briefing on what we've got:

* **Core**
  + **Engine** – the central unit of the system
  + **CollisionHandler** – called to process collisions between objects in the system
  + **ConsoleRenderer** – responsible for rendering graphics on the console
  + **ObjectGenerator** – generates objects for the engine, you will insert your test objects here whenever you want them to appear in the environment
* **Interfaces –** the followinginterfaces define the behavior (methods) and properties of:
  + **IObjectGenerator** - produces a set of objecets that are to be added to the environment system
  + **ICollidable** – objects that can collide
  + **ICollisionHandler** - defines behavior for managing collisions of **ICollidable** objects
  + **IObjectProducer** – objects that may produce other objects
  + **IRenderable** – objects that may be renderered to the screen
  + **IRenderer** – describes what a renderer should do
  + **IController** – defines a controller (e.g. mouse, keyboard, touch screen) with a set of **events**, will discuss later
* **Models**
  + **Objects**
    - **EnvironmentObject** – base class for all objects in the game, implements ICollidable, IObjectsProducer, IRenderable
    - **MovingObject** – inherits Environment object and also holds Direction of movement
    - **Ground** – a ground object
    - **Snowflake** – a snowflake object
  + **Data.Structures**
    - **QuadTree** – a tree-like data structure used for collision detection optimization
    - **Rectangle** – represents a rectangular entity, used to describe the 2D position and size of objects

Analyze the described classes more thoroughly. Understand how the **EnvironmentObject** and **his descendants** work and how the **engine uses them**.

…and don't forget to run your project!