# Розробка ігрових застосувань

## Лекція 2

**Основи комп'ютерного рушія та складових елементів гри**

**Додаток для початківців у мові програмування C#**

# Dedicated to

**Adam Freeman** is an experienced IT professional who has held senior positions in a range of companies, most recently serving as chief technology officer and chief operating officer of a global bank. Now retired, he spends his time writing and long-distance running.

# «Nullius in verba»

# Література базова

1. https://unity.com

2. C# 4.0:  The Complete Reference Herbert Schild

# Література допоміжна

https://blog.studica.com/how-to-setup-github-with-unity-step-by-step-instructions

https://unlix.ru/%D0%BE%D1%88%D0%B8%D0%B1%D0%BA%D0%B0-warning-lf-will-be-replaced-by-crlf-%D0%B2-git-%D0%BA%D0%B0%D0%BA-%D1%80%D0%B5%D1%88%D0%B8%D1%82%D1%8C/

# Питання лекції

1. Introducing Classes and Objects

2. Створення 2D проекту, підключення інструментарію та знайомство з Asset Store.

3. Базовий патерн ігрового рушія

# 1. Class Fundamentals

```
class classname {
 // declare instance variables
access type var1;
access type var2;
 // ...
access type varN;
 // declare methods
access ret-type method1(parameters) {
 // body of method
 }
access ret-type method2(parameters) {
 // body of method
 }
 // ...
access ret-type methodN(parameters) {
 // body of method
 }
 }
```

Notice that each variable and method declaration is preceded with access. Here, access is an access specifier, such as public, which specifies how the member can be accessed.

# 1. Define a Class

The first version of Building is shown here. It defines three instance variables: Floors, Area, and Occupants. Notice that Building does not contain any methods. Thus, it is currently a data-only class. (Subsequent sections will add methods to it.)

```
class Building {
 public int Floors; // number of floors
 public int Area; // total square footage of building
 public int Occupants; // number of occupants
}
```

To actually create a Building object, you will use a statement like the following:

```
Building house = new Building(); // create an object of type building
```

# 1. Define a Class

The first version of Building is shown here. It defines three instance variables: Floors, Area, and Occupants. Notice that Building does not contain any methods. Thus, it is currently a data-only class. (Subsequent sections will add methods to it.)

```
class Building {
 public int Floors; // number of floors
 public int Area; // total square footage of building
 public int Occupants; // number of occupants
}
```

To actually create a Building object, you will use a statement like the following:

```
Building house = new Building(); // create an object of type building
```

# 1. A program that uses the Building class.

```
using System;

class Building {
 public int Floors; // number of floors
 public int Area; // total square footage of building
 public int Occupants; // number of occupants
}

// This class declares an object of type Building.
class BuildingDemo {
 static void Main() {
 Building house = new Building(); // create a Building
object
 int areaPP; // area per person
 // Assign values to fields in house.
 house.Occupants = 4;
 house.Area = 2500;
 house.Floors = 2;

// Compute the area per person.
 areaPP = house.Area / house.Occupants;
 Console.WriteLine("house has:\n " +
 house.Floors + " floors\n " +
 house.Occupants + " occupants\n " +
 house.Area + " total area\n " +
 areaPP + " area per person");
 }
}
```

# 1. A program that uses the Building class.

This program consists of two classes: Building and BuildingDemo. Inside BuildingDemo, the Main( ) method creates an instance of Building called house. Then the code within Main( ) accesses the instance variables associated with house, assigning them values and using those values.

The program displays the following output:

```
house has:
 2 floors
 4 occupants
 2500 total area
 625 area per person
```