Classes

THEORY

Classes are a huge topic, so brace yourself, because I’ll talk a bit. First a bit of theory: C# is a so-called Object Oriented Programming Language, fancy words uhu, this means that when you program in C# you should think in objects. E.g. if you have a program that deals with cars you’ll want to create a car object and maybe give it some properties (max speed, color, model,…).

Classes allow you to do just that and much more! A class is basically a definition of an object that can have properties and can perform actions, in developer-language properties are variables and actions are methods.

When you created a definition for an object you’ll want to create an object that follow that definition, in our car example this would be creating the cars. The “car” object will be stored inside a variable. As you understand we can have many cars that follow the same model, but have different properties (variables).

PRACTICE

So here is the syntax to create a class

class className{

//things inside the class

}

pretty easy, right?

When you want to create an object that follow the definition (called instance of the class), you can use this syntax:

className()

This will return an object (it’s a method, did you recognize the syntax?).

So our car model would work out like this:

class Car{

float speed;

string color;

string model;

}

\\ ….

var ferrari = Car();

Now that we have our ferrari object, we want to assign its variables (or use its methods), we can use the syntax

className.Property

so for example to set the max speed of the ferrari to 350 km/h

ferrari.speed=350;

Let’s suppose that every car cost 200 times his max speed and we wanted to create a function that gives us the cost of the car we need to edit our like this:

class Car{

float speed;

string color;

string model;

float price(){

return speed\*200;

}

}

\\ ….

var ferrari = Car();

ferrari.speed=350;

float price=ferrari.price() //price has a value of 70000 (350\*200)

THEORY

Second piece of theory, here it comes: C# has this thing called accessibility modifiers. Basically they decide whether a method/variable is visible from outside the class or not.

The most important modifiers are public and private, as you can guess public means it’s accessible from other classes, private means it’s not.

PRACTICE

To use those modifiers you just need to put them before the declarations

e.g.

public int x;

public void newMethod(){

}

private int y;

private void newMethod2(){

}

Easy right?

ASSIGNEMENT

So let’s try and make something with all these “classes thing”.

We’ll create a more complex program this time, so don’t worry if it takes longer to figure it out.

We’ll create a very simple two-players game, the rules are:

* every player has 3 stats: health, armor, damage
* every player can perform 2 actions: attack or improve one stat by one point
  + when attacking, the other player loses n health points, where n is the difference between the attacker damage and the defender armor (if the defender has more armor than the attacker damage, he isn’t damaged, but doesn’t heal either)
* you lose when you reach 0 health points

Interactions:

* to attack type “attack”
* to improve a stat type “improve statName”

Every turn goes like this:

* the current stats for each player are displayed
* the text “Player 1 turn” is displayed
* the player 1 types his action
* the text “Player 2 turn” is displayed
* the player 2 types his action

When a player loses the text “Player X won” is displayed