Programming Assignment #2

Fantasy game battle simulation

Start with the accompanying worksheet and write a small library (.dll) for the role playing game interfaces ICharacter and IAttack. The library will support various types of fantasy characters and their different attack types. The class hierarchy will look like the following:

- ICharacter
  - CharacerBase
    - Archer
    - ComputerWizard
    - Mage
    - Warrior
- IAttack
  - NormalAttack
    - BadGradeAttack
    - BowAttack
    - FireAttack
    - SwordAttack

ICharacter and IAttack are interfaces. Your sample application below should only create variables of these types for use in referencing characters and their attacks. This is called "programming to interfaces", and it is a preferred design strategy to promote code reuse.

CharacterBase and NormalAttack are base class implementations of ICharacter and IAttack respectively. We don't expect to instantiate them directly, but they each provide some default behavior for their interfaces. Note that this design isn't required and each of the character classes and/or attack classes could have directly implemented their respective interfaces without having a base class as a parent. Such a design might be more appropriate where no default implementation makes sense.

Archer, ComputerWizard, Mage, and Warrior are the specific types of characters our battle simulation will model. Likewise, BadGradeAttack, BowAttack, FireAttack, and SwordAttack are their attacks.

Worksheet 2 helped you implement and test a few of these classes. You should move all your implementations into the library, and implement the rest of the classes above.

To complete your assignment, you need a more thorough combat system which determines such things as who is in which party; who attacks first, second, etc.; who each character decides to attack, and when the combat is over. I have provided an implementation of this as a class file you can download here: Combat.cs

Go ahead and change the namespace to match your own, or add a using directive to the top of the file, so it can reference your ICharacter and IAttack interfaces. There is one public class, Combat, which has one public method AutoBattle and several other methods to perform the battle steps we described above.

Let us go ahead and extract an interface from this class for use in our sample application. Right-click on the class Combat and choose Extract Interface. A dialog comes up and choosed the default name of ICombat for the interface name. Hit Select All to make sure the AutoBattle method is included in the interface. Hit ok.

Sample Application

To test your library, you need to create a player party and enemy party (each of type IList<ICharacter>) and use them to instantiate one Combat. Finally, call AutoBattle to perform the battle simulation.

Try and tune your battle simulation so the player party wins the battle about half the time. You can do this by modifying the health/number of combatants. Don't give them too much health, taking too long can hinder testing.

Additional Tasks

1. In the Properties folder, Open the AssemblyInfo.cs file and fill in the Title through the Copyright information. This information will be displayed if you look at the properties of your .exe file. Be careful messing around with the assembly though, it can be very easy to break your project this way.
2. Provide comments explaining your logic. Also, add a block comment to the beginning of the file listing your name, the course and a description of the lab (from above but in a completed tense).
3. Do a Build Clean, zip up your solution (no .exe files are allowed) and then email your completed assignment to me.