LaSalle College
Vancouver

VGP 232 Game Tools and Pipeline

Assignment 2a

Console Application (improvement)

Objective:

Using your existing Assignment 1 **C# Console Application** improve it by refactoring:

Getting Started:

- Creating the **Assignment2a** project from Assignment1
    a. In Visual Studios: Project > Export Template
    b. Check “Project template” and Select “Assignment1”
    c. Change the Template name to **Assignment1Temp** and press Finish
    d. Right Click the solution in the Solution Explorer panel
    e. In the context menu, select Add > New project...
    f. In the search box, type in Assignment1, and select it from the list and press Next
    g. Change the Project name to **Assignment2a**
    h. Copy the IPersistence.cs, UnitTests.cs and data2.csv in Assignment2a
    i. Remove the data.csv
    j. Set the data2.csv to always copy in the Properties panel.
- Add the unit test framework using NuGet package manager
    a. Right click the Assignment2a in the Solution Explorer
    b. Select Manage NuGet Packages…
    c. Switch from the Installed tab to the Browse tab
    d. Install Micrsoft.NET.Test.Sdk latest version 16.7.1
    e. Install NUnit latest version 3.12.0
    f. Install NUnit3TestAdapter
- Running the tests
    a. Double Click on Assignment2a
    b. Paste <GenerateProgramFile>false</GenerateProgramFile> within <PropertyGroup> between line 4 and line 6.
    c. Test > Test Explorer (Ctrl + E, T)
    d. Click on the Play button (Run all Tests in View (Ctrl + R,V))

Refactoring

- Extend the Weapon class (i.e. Weapon.cs)
  Implements the following methods:

VGP 232 Game Tools and Pipeline

    - Add the following properties: Image (string), SecondaryStat (string), Passive (string)
    - Add a **public enum WeaponType** with the following values: **Sword, Polearm, Claymore, Catalyst, Bow**, **None** and change the existing Type property from string to WeaponType
    - **static bool TryParse(string rawData, out Weapon weapon)** should similar to float.TryParse(string, out float)
    - Update the **ToString()** method so it can print the **CSV (comma separated value)** format with the new properties
- Add an interface IPersistence (IPeristence.cs)
    - bool Save(string filename)
    - bool Load(string filename)
- Add a WeaponCollection class (WeaponCollection.cs) which inherits from List<Weapon> i.e. public class WeaponCollection : List<Weapon>
  Implements the following methods:
    - int GetHighestBaseAttack()
    - int GetLowestBaseAttack()
    - List<Weapon> GetAllWeaponsOfType(WeaponType type)
    - List<Weapon> GetAllWeaponsOfRarity(int stars)
    - void SortBy(string columnName)
    - Inherit from IPersistence above

Refactoring:

- In your main method, replace the **List<Weapon>** results with the class **WeaponCollection**
- Move the static **Parse** function out of Program.cs main method and invoke **Load** function from WeaponCollection class
- The **Weapon** class will have a TryParse(...) mentioned above which Load will call and handle any exceptions that may occur when the values that are being parsed is invalid i.e. invalid number of rows, wrong types to convert etc.

Unit Test:

Add nUnit tests to confirm that your class can do the following (included in UnitTests.cs):

- WeaponCollection
    - Get the highest base attack value
    - Get the lowest base attack value
    - Get all weapons of a certain type i.e. Sword
    - Get all weapons of a certain rarity i.e. 4
    - Load a valid data
    - Load an invalid data

Revised: 10/15/2020

VGP 232 Game Tools and Pipeline

        - Save the csv file output
            - Note for this, you may need to use the SetUp and TearDown step so it runs it cleanly
    - Weapon
        - TryParse valid values
        - TryParse invalid values

Error Handling:
- invalid path (path does not exist)
- invalid arguments
- invalid data from the csv

C# Helpful links

https://support.microsoft.com/en-us/help/816149/how-to-read-from-and-write-to-a-text-file-by-using-visual-c

https://www.dotnetperls.com/sort-list

Due date

Next class, week 3 (at the start of class 6:30pm)

Submission

Implement the answers in a new project with the name “Assignment2”. This project should be submitted through committing and pushing through GIT to your VGP232 repository which you shared with the instructor.

Grading

Marks: Out of 100 (10% of final grade)

(60) Functional: Does it compile? Does it meet the requirements and work? Does it give the correct results?

(25) Error Handling: Does your application handle bad input? If so, does it handle failures and exceptions gracefully or does it crash?

LaSalle College
Vancouver

VGP 232 Game Tools and Pipeline

(15) Naming convention & Comments: Does it follow the coding standards? Are your variables and method names descriptive? Did you leave descriptive comments on methods that does not have obvious functionality?

Coding Standard

C#
https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions

Late Assignments

After the due date, the student will no longer be able to submit their assignment and will receive a 0 as I will go over the solution in the next class.

Revised: 10/15/2020