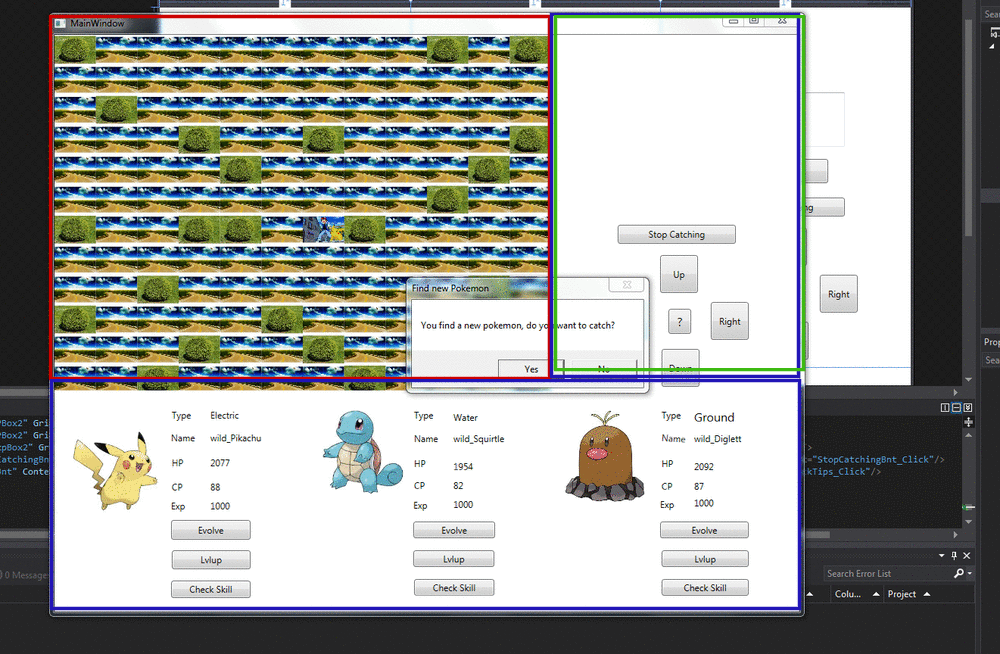
Project Report---Pokemon Go

-- By Huang Cen (1155062136)

Li Zihao (1155077069)

1. Introduction
2. Overall introduction:

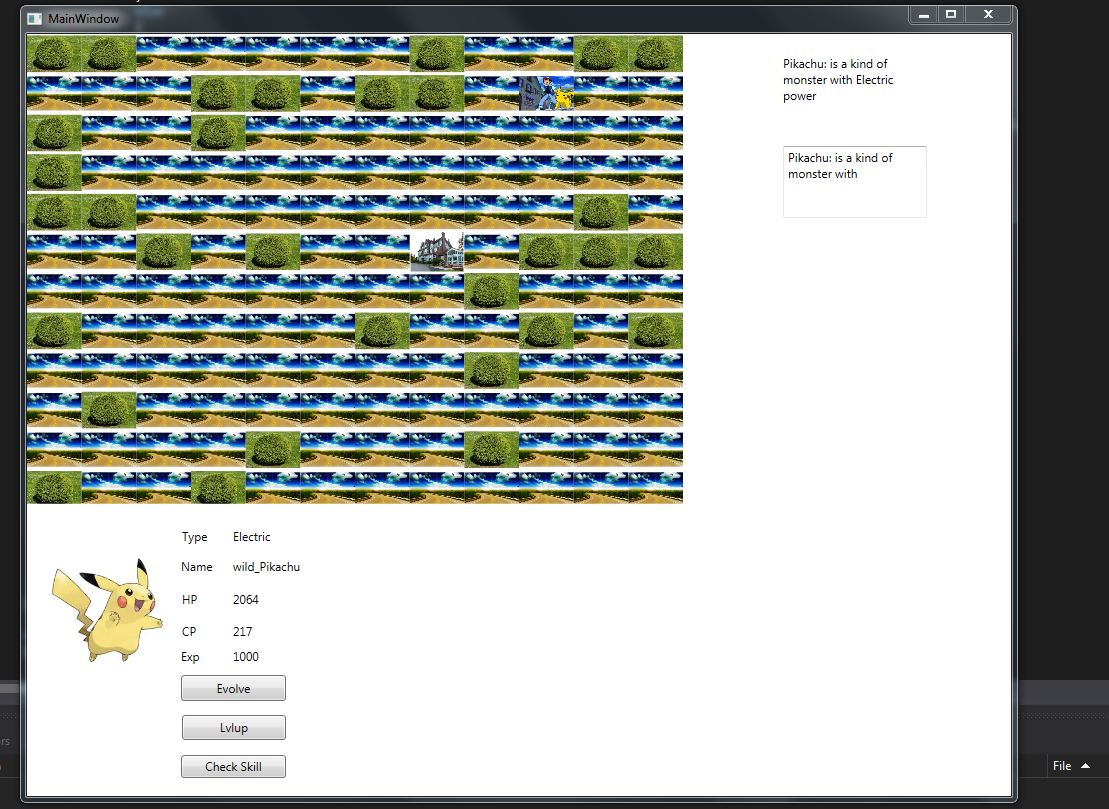
In our project, we have designed a game which provides players with multiple tastes with navigating around the map, capturing pokemons, waiting for pokemons to accumulate exp value and get level up, as well as battling in the gym with NPC. Although it’s an off-line, single-player game, it adopts MVP software architecture and separates model and view in a sense. Also, our game combines “Wu Xing” concept from Chinese philosophy and attacking mechanism from <League of Legend>, a famous MOBA playing game played all over the world these days.

1. Introduction of development environment

Basically, we are using Visual Studio 2013 as the developing tool, adopt C# programming language as the main developing language and use the WPF as the developing interface.

The whole user interface is mainly developed into 3 parts: Navigation Map District (red), Pokemon checking District (blue) and Manipulating & Presenting District (green). First of all, in the navigating part, players are supposed to wander around to find wild pokemons, however, except from the normal roads, we also add some bushes to forbid the players to pass through. In this way, the difficulty of finding a pokemon is increased. There may also have some gyms for players to combat. In these gyms, players can choose different difficulties to combat with different levels of NPCs. In the pokemon-checking area, players can get the real-time information and check the states of the pokemons. As you can see, we only have three pokemon balls, thus players can only have three pokemons in one trial. If one player has caught a new pokemon when the three balls are filled, he either needs to drop one of pokemon in the ball or give up the new pokemon. In the manipulating and presenting area, players can use the buttons to choose the navigating direction, get the tips for how to play the game, or even see the capturing & combating scene! We will illustrate the capturing and combating in the later parts.

1. Introduction of the Capturing

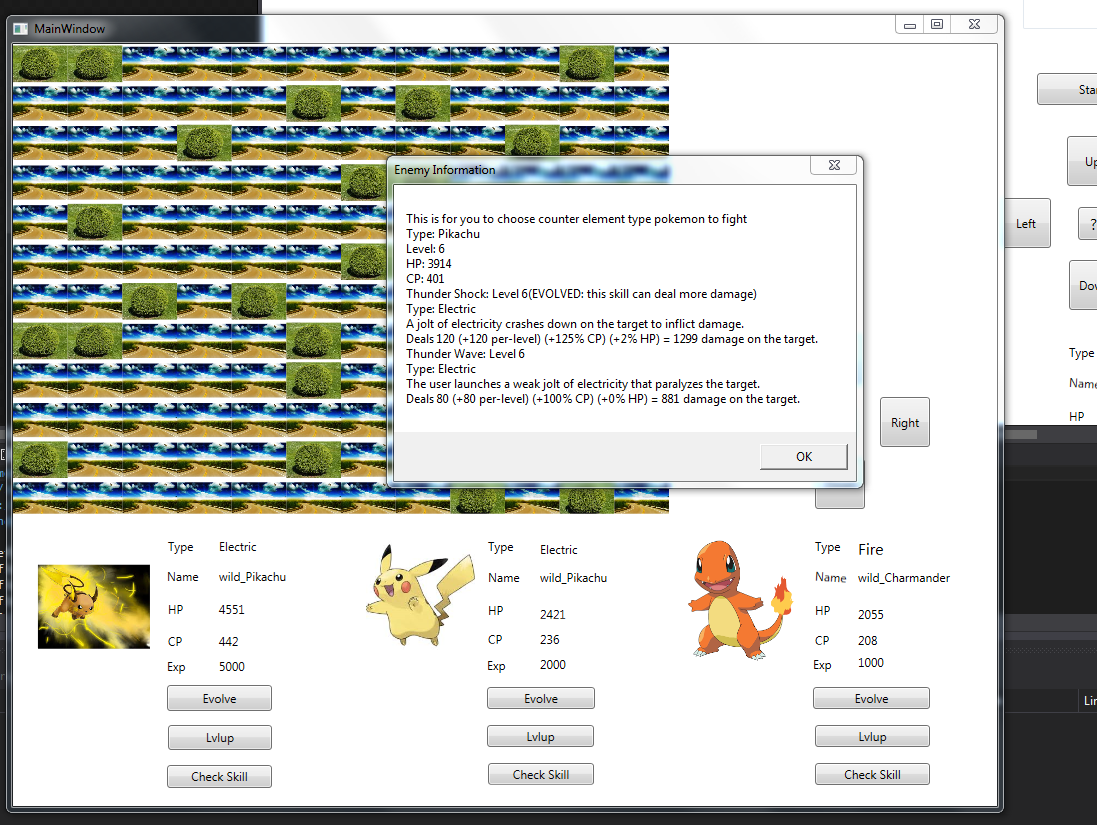


Player would be given randomized pokemons when they are navigating around the map, in this part player needs to do fast-typing to catch the pokemon, once the pokemon is caught, the information file would be shown in the checking area.

1. Introduction of the Combating

The procedure can be listed as below:

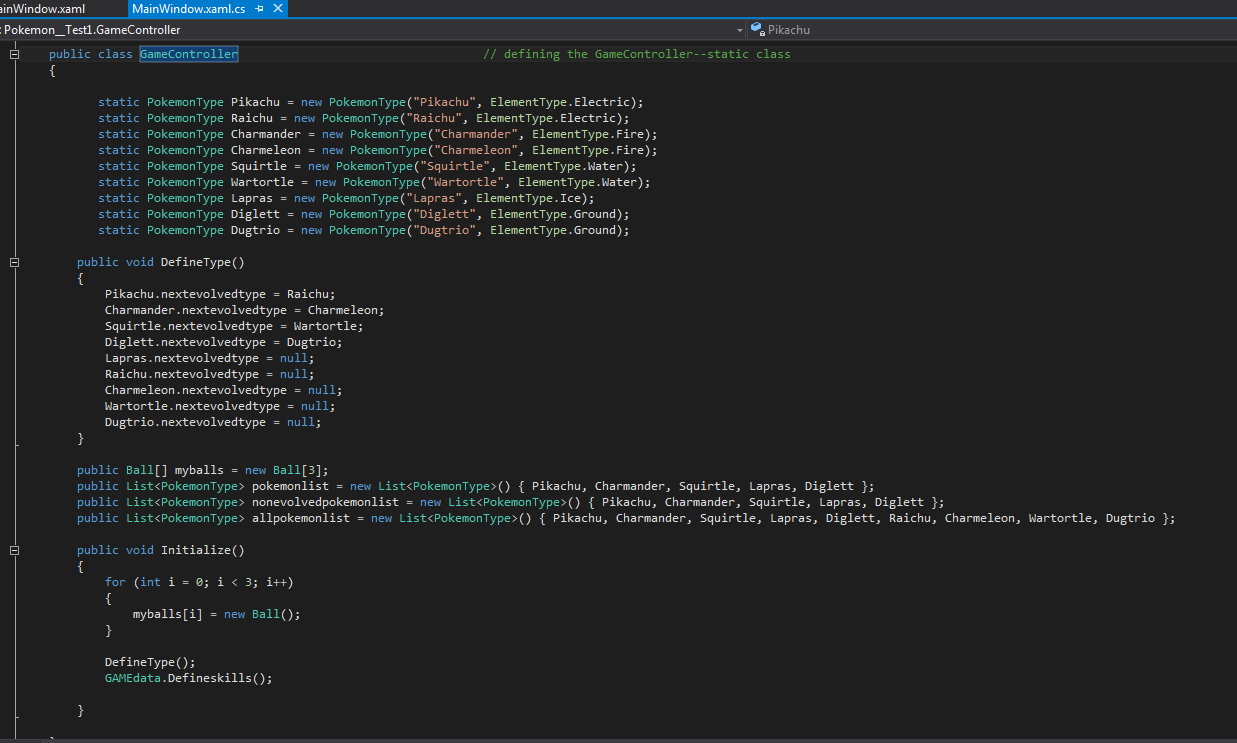
Go to the gym—Choose the difficulty—Check the rival’s information and choose the pokemon to combat—Choose the skill to attack—Wait for the other side to attack--…--Win/Lose



1. Class design
2. GameController:

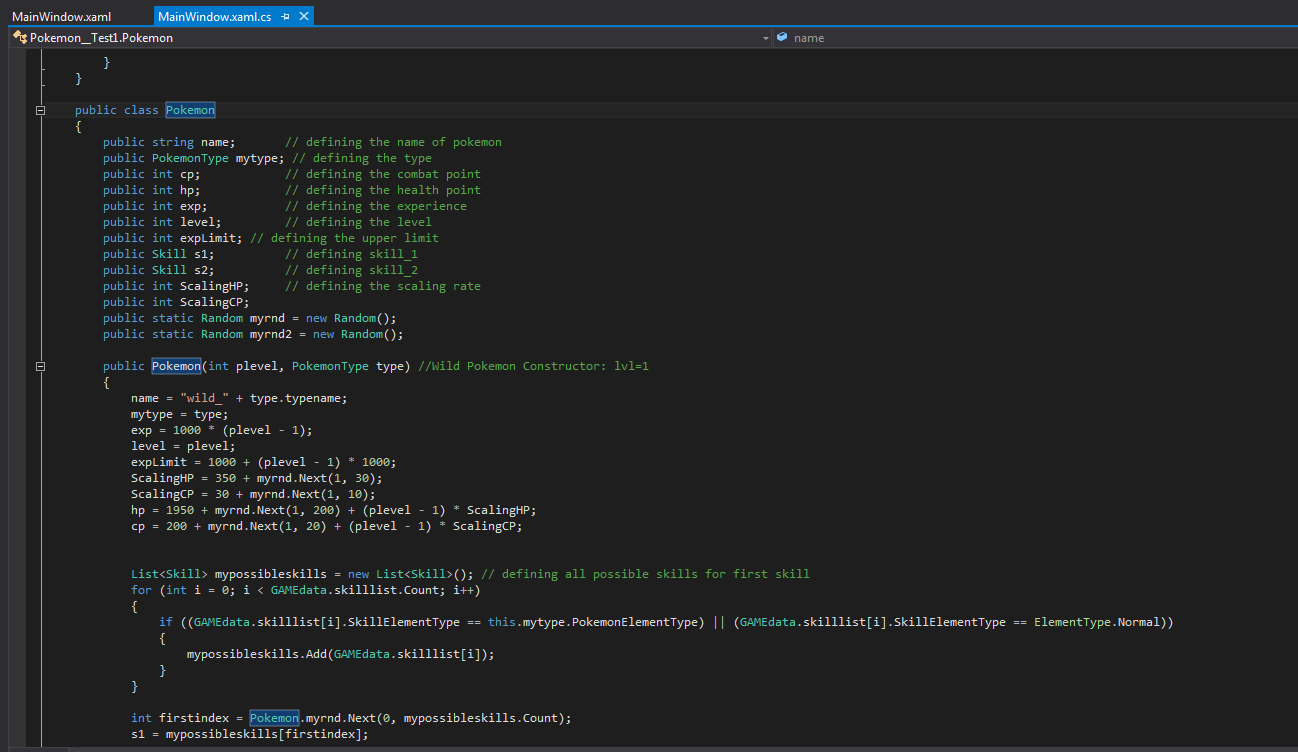
For the GameController class, we mainly design the basic pokemon types and evolved pokemon types as static members, initialize the 3 balls in the player pockets.

As this is a single-player and off-line game, we adopt the *singleton design* pattern for GameController class.



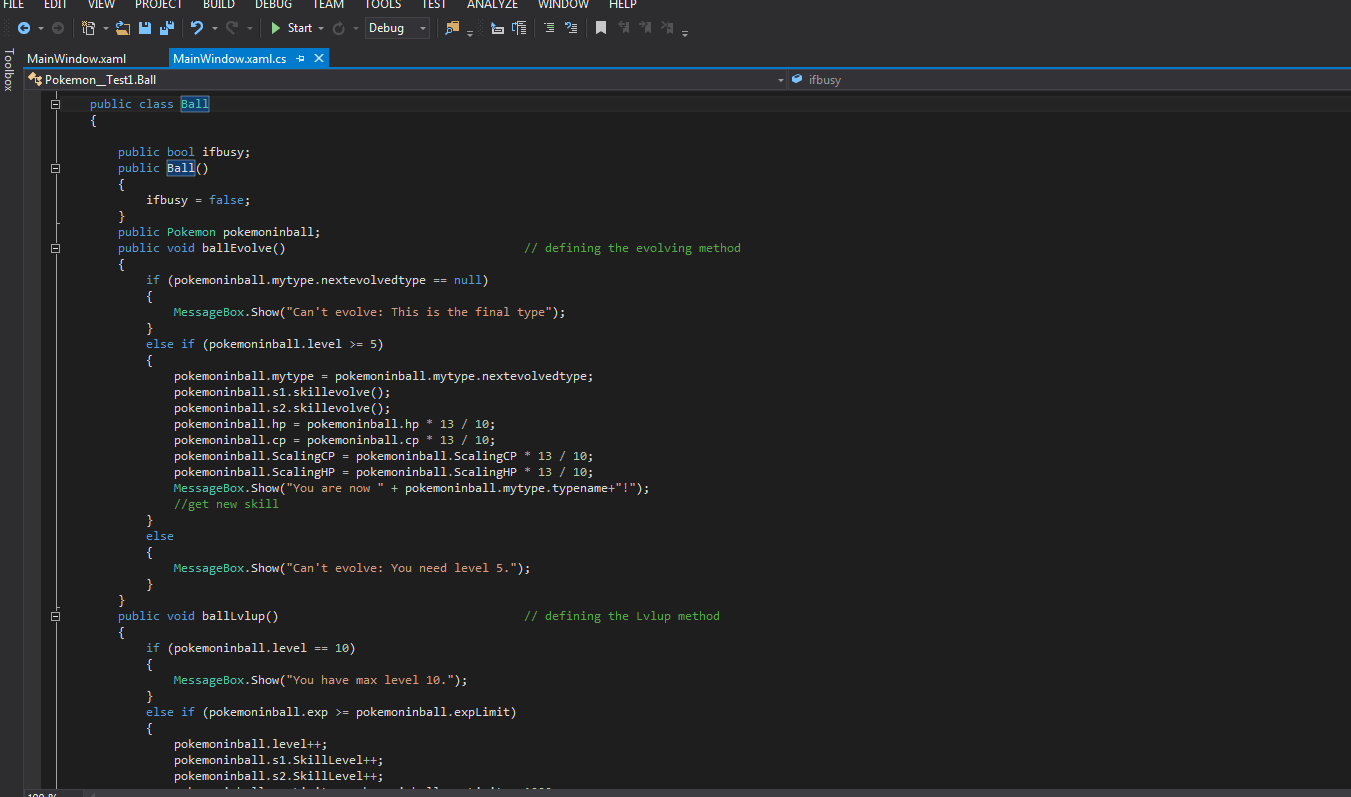
1. Pokemon & PokemonType:

In our developing process, we assign each pokemon with multiple fields such as name, HP, CP, exp and two skills to make the battle more exciting. Actually, we also design a *subclass* called PokemonType which is composed with typename and Elementtype. For instance, we have five fixed elemental pokemon types: Pikachu (electric element type), Squirtle (water element type), Chamander (fire element type), Lapras (ice element type) and Diglett (ground element type). There are two custom constructors named “wild constructors” and “NPC constructors”. We will use wild constructor to randomly create a new pokemon for you to catch, however, when you choose the different levels of difficulty, the program will automatically use the NPC constructor and put in different parameters to create different pokemons.



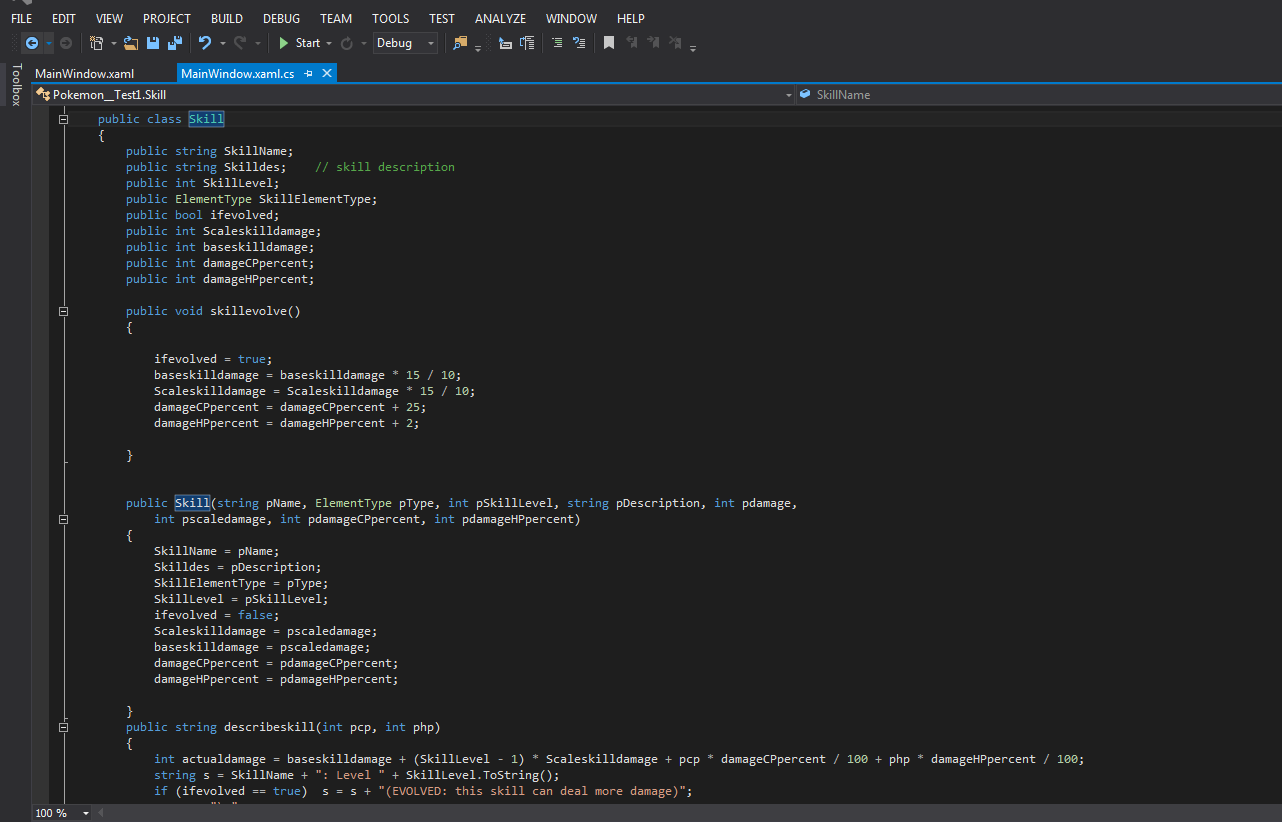
1. Ball

We choose to design a Ball class to store the pokemons for two reasons: First of all, we want to restrict the number of pokemons one player can have in one trial to avoid explosion. Secondly, we want to use the Ball class as the *adaptor between model and view*: In the model part, we encapsulate the logic of level-up and evolving from the user (these two methods are put in pokemon class, Ball class simply calls them). The model cannot know how the pokemon is presented in balls. However, in the view part, it just use text blocks and buttons to manipulate the changing of balls, it cannot know how the pokemon is evolved or even don’t know what’s inside the ball!



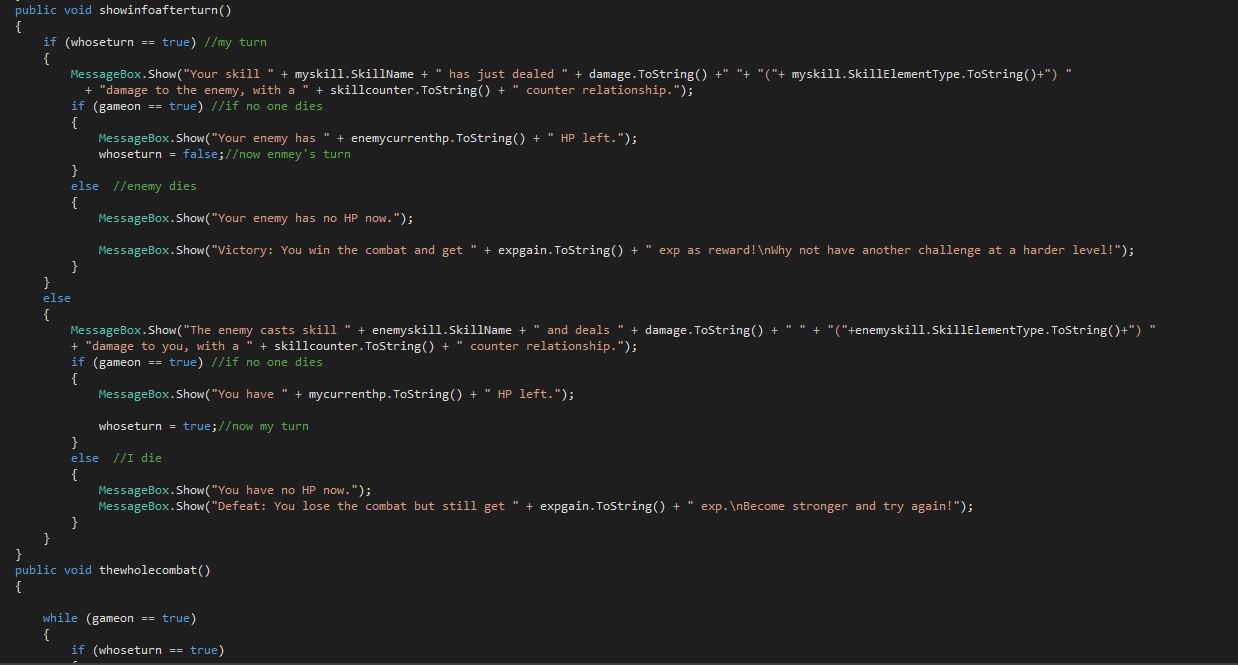
1. Skill

The skill class applies the software design pattern –*strategy pattern* and uses the design concepts of MOBA games like LOL. First of all, like the pokemontype class, the instances of skill class are created in the initialization function, which means the skills are limited and predefined. When we create the pokemon, the instance would randomly choose two skills from the skill set corresponding to the element type. Secondly, although there are finite amount of skills to form a skill set (a global variable), the damage would increase as the pokemon’s level become higher. The pokemon level up function would *callback* the skill level up function automatically. In this way, our design team further *encapsulates* the logic from the outsiders. Finally, in order to make our class more *user friendly*, we add a method called skilldes() to illustrate the concrete functions of skills. Skill class is very important to form the functionalities of combating procedure.



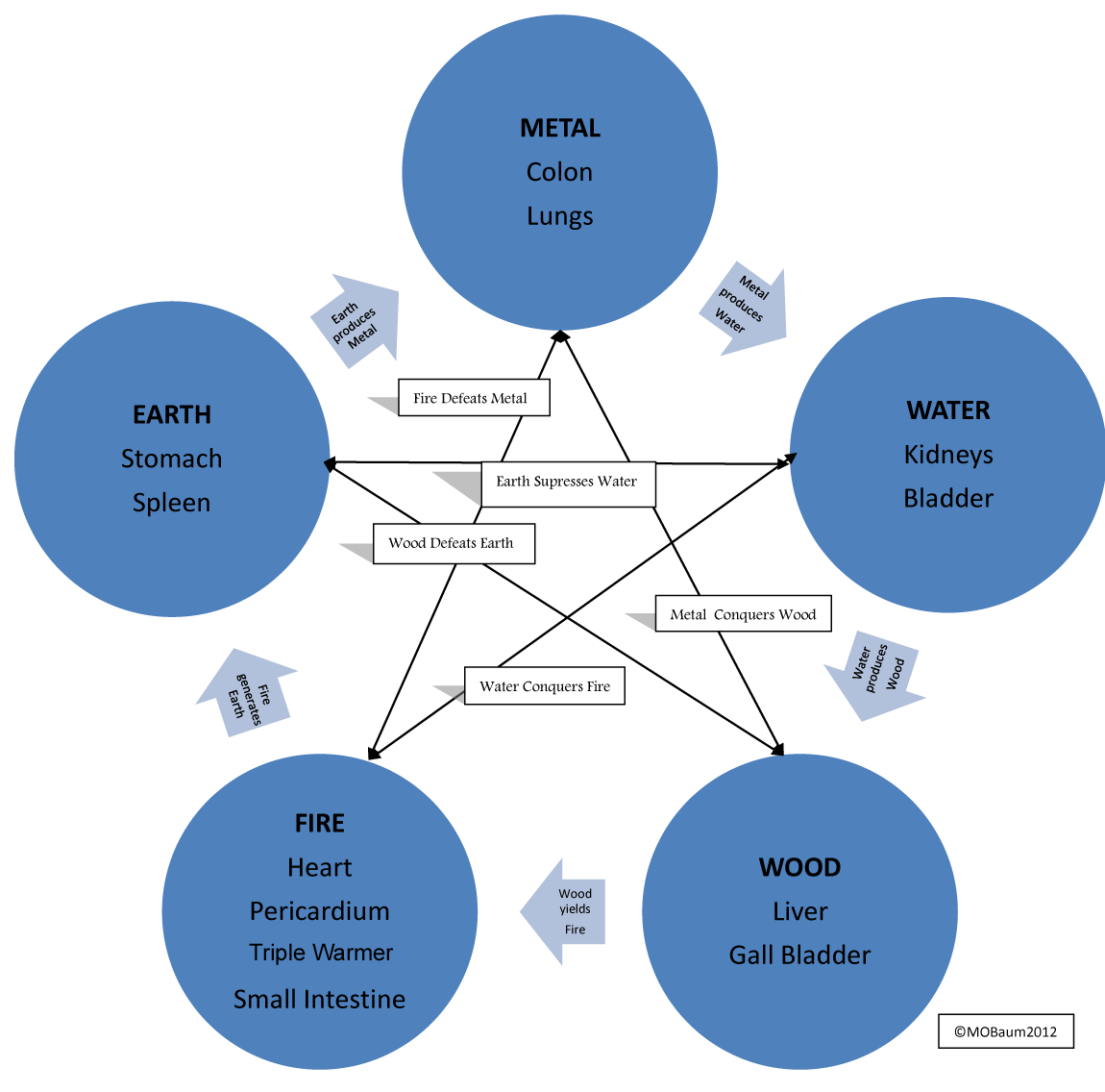
1. Combat

The combat class is based on the skill class, it acts more functional so that we’d like to regard it as *Façade pattern*. I’d like to introduce the procedure of using this façade: first of all, the view part would ask the user to input the difficulty and the pokemon-id (one of three balls). Then the model part would use the difficulty to randomize a NPC pokemon. Then the two pokemon would be put into a method that each one of them takes turns to attack the other until one of them has HP<0. If it’s in the player’s attack, the program would ask for one of the two skills to be used to attack, if it’s in the NPC’s attack, the NPC would randomly choose a skill to attack. After one turn of attacking, the damaged would be calculated to find whether one’s HP is under 0. If not, the loop would go on. If so, it would jump out of the loop and gives exp to player according to the game result. (If win, the player would get more exp)



1. Software patterns

According to the previous descriptions, you will find a lot of software patterns are used in the process of class design. To put it in a nutshell, we’ve applied *Singleton* from the *creational pattern* for the GameController, *adapter pattern* in the Ball class, *strategy pattern* in the Skill class and *Façade pattern* in Combat class. Overall, we adopt the MVP software architecture and separate the model and view as far as possible. Also in our designing process, we increase the combating complexity by inserting some creative thoughts. For example, we use “Wu Xing” concept from Chinese philosophy to model the countering relationship between elements: Ground cons Electric, Electric cons Water, Water cons Fires, Fire cons Ice, Ice cons Ground. So the value of damage has been carefully calculated by our group in terms of the element relationship.

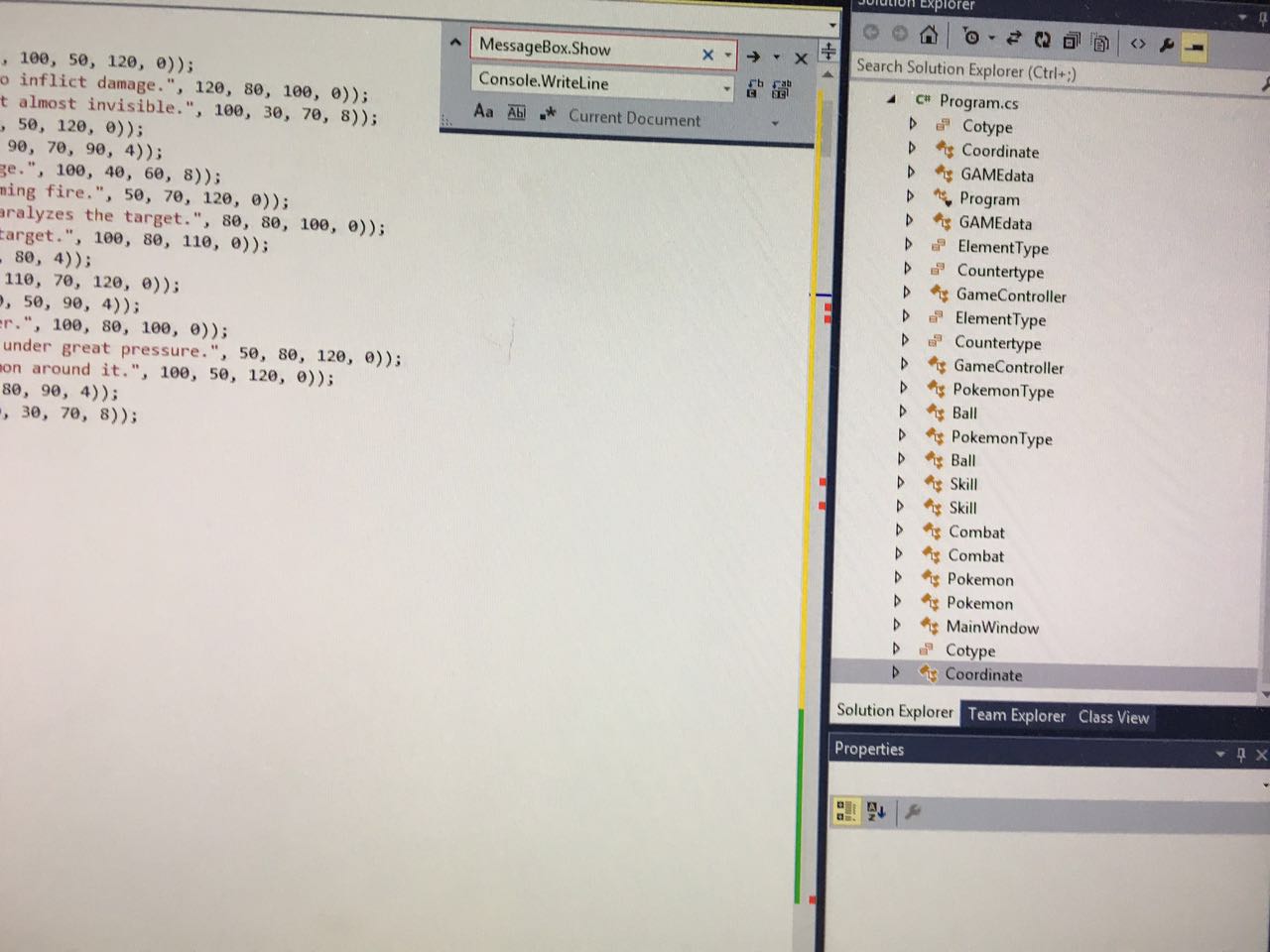


1. Difficulties overcame in our project

There are a few difficulties we had overcame in our project development process. First of all, some of the class are really misleading because of the deep hierarchy and similar names. (like Pokemon and PokemonType) Later we wrote some UML to illustrate the relationship between different classes and find out how to use them. Secondly, we have difficulties in the data design because in our design patten, the calculation is too complicated that we will easily lose the balance of game. (e.g Some pokemons have too strong skills and can easily beat others) In this part, we did a lot of investigation in the data center and carefully revise the data by BlackBox Testing. In the end, we got the appropriate data and applied them into use.

1. Class reusing

Apart from the GameController class, we almost reused all the classes in the Console version. Actually, we combined the GameController class into the Program class and delete it. For the other classes (Ball, Pokemon, Skill,Combat), they all have good model-view separation and can be reused easily.



Q&A from the designer

1. What’s the work division of your group?

Answer: Our group is composed of two person:

Huang Cen (Richard) SID: 1155062136

Li Zihao (Jack) SID: 1155077069

Huang Cen deals with (1) the whole view part, including the GridMap initialization, the whole capturing part (fast-typing), all of the controls (Buttons, text blocks, images interchanging), the pokemon timer and the exp\_timer. (2) The class design of GameController class , Pokemon class (3) the lab report

Zihao deals with (1) the Ball class, the whole Skill class as well as its data investigation and calculation using BlackBox Testing, the Combat class as well as its gaming mechanism (2) the whole reusing class procedure and the Text Mode programming.

1. What if I am not familiar with the game and don’t know what to do?

Answer: Do you notice there is a button in the middle of the direction controller?

It is labelled with a “?”. If you click this button, it will randomly give you some hints for the game.

1. What if I already have three balls filled and still have chance to get a new pokemon?

Answer: Our design is that you can still get chance to catch more pokemons even the balls are all filled. There are two choices for you: If you don’t want anymore pokemons, you can simply refuse the catching by clicking “no”. However, if you are not satisfied with the existing pokemons in your balls, you can still have a chance to replace them with new pokemons.

1. What if the rivals’ pokemons are to strong and I cannot beat them?

Answer: There are many reasons if you lose the game: Maybe the rivals are countering the element of your pokemon, so it’s your freedom to choose the right pokemon to fight. Secondly, it maybe your fault to choose the wrong skill to attack, always look at the skill instructions by clicking “Check Skill” Button. Thirdly, maybe your pokemon hasn’t been evolved after reaching level 5, remember after evolving the pokemon would have a large improvement in HP, CP, skill\_damage and all kinds of properties. Finally, it may be due to the low level of your pokemon, so in the end, spend more time with your pokemons and love them! BTW, even if you lose the battle, you will still get some exp from it, so never lose your faith. Always keep fighting, summoner!