Object Oriented Analysis and Design

Sema Dilber

November 2021

1 Campuses and Puzzles

1.1 Abstract Factory Design Pattern

Actually, I had difficulty in designing due to the complexity of both the pattern and the structure in the assignment, and some things did not suit me, but I benefited from the pizza example in the book,again and I think they all look the same. I don't like to do exactly the same example in the book, but because it's my first time, I'm following this path..
By looking at the structure in the book, I can explain what I think and why I use which structure in a comparative way: Since the main issue that separates the Abstarct factory from the factory is the ingredients issue, I first thought about what could be equivalent to the ingredients in the example in the homework. Here, Strength, Health and Agility properties, which are valid for both style and type, appeared equivalent to properties such as Dough-Sauce, and x1.3, x0.8 properties appeared to be equivalent to properties such as ThickCrust-Dough. That's why I created interfaces such as Strength, Health and Agility and derived subclasses like 1.3 (with methods that return these values). (I know that classes that only generate a value aren't nice and correct)

Then, I created IngredientFactories for the styles given in the assignment (Atlantis, Valhalla, Underwild) just like PizzaIngredientFactory and NY/Chicago Pizza IngredientFactory, and I created the appropriate strength, health and agility for these styles. (Such as AtlantisPowerIngredientFactory createStrength returns new Strengthx08())

In addition, I extended Red, Blue and Green from my Power class, which are equivalent to the types in the book, and created the pizzas in accordance with the types in the abstract prepare() class in Power, and I used ingredient factories in this way.

Also, There are the PowerStore, and ValhallaPowerStore etc. use derived from it are that I created as unnecessary or failed to. I didn't use them but still didn't want to delete them. I tried to create an equivalent to PizzaStore in the

book, but I was very confused. Maybe that wasn't unnecessary one.

1.2 The Game and What to Say About It

While making the board and movements of the game, I benefited from the candy crush style games available on the internet. In general, the game works. There are randomly generated Enemys above and current Health levels are written. Each time they take damage, this level decreases and when this level falls below 0 for all 3 enemies, Enemys die and are recreated. (It happens with the action right after the health level drops below 0.)
I created a char, but I didn't put it in the game because I didn't use it. Only one player can play against Enemys(E1,E2,E3 met with enemies[3] array), the enemy does not produce a random counter move. However, the matches made by the player (and accordingly other matches in the table) will damage the enemies. In order to realize the phrase in the PDF that it will damage the enemy it crosses: I made the first star in the match damage the enemy with whichever monster it intersects. I also provided the damage rate said in the PDF. Since I could not obtain only the colors of the stars, I created any random number instead of those stars and gave the result what would be the result if the star was that color. So only if the real color is reached from the star image, I will have achieved what is requested.
Another thing is that although I didn't use C1,C2,C3 characters, I created them (chars[3] array) and brought them to the level where they can use everything the enemies use. I also made conditions such as creating randomly at the beginning, ending the game when the Health level drops below 0. In case a random code is generated, the code is suitable for enemies to move and damage.
Also, an important point is that in all damage, the required enemy is damaged at the right rate, but sometimes too much damage occurs due to consecutive matches. The damage written on the screen is only the damage given by the player with a single move. The large reductions in damage are due to this.

You can look at to my code, javadoc and class diagrams for more details.
Requirement
Java version 16 is used, 16 is required for the jar file to work.