PrimordialSoup: A Cook's Tour

Note: this article is based on PrimordialSoup 5.2.x.

Introduction

In an earlier task (see problemset02: Snake and Ladder), we implemented another game. In this task, we will take a peek under the covers and show you how the Primordial Soup itself is constructed.

We open with a discussion of the goals of the game. The goals will reappear in many small details during the presentation of the game itself. Following this, we present the design and implementation of the game. The design will be described in terms of patterns (surprise, surprise), the implementation as a literate program.

Goals

What are the goals of PrimordialSoup?

There will be three players that can participate in the game. Everyone starts with one amoeba and can have at most seven. Every round they drift on the board and eat some food and can buy a gene with special abilities if they have enough biological points. The more amoebas a player has the more points he gets and the further ha can move up the ladder.

The Design of PrimordialSoup

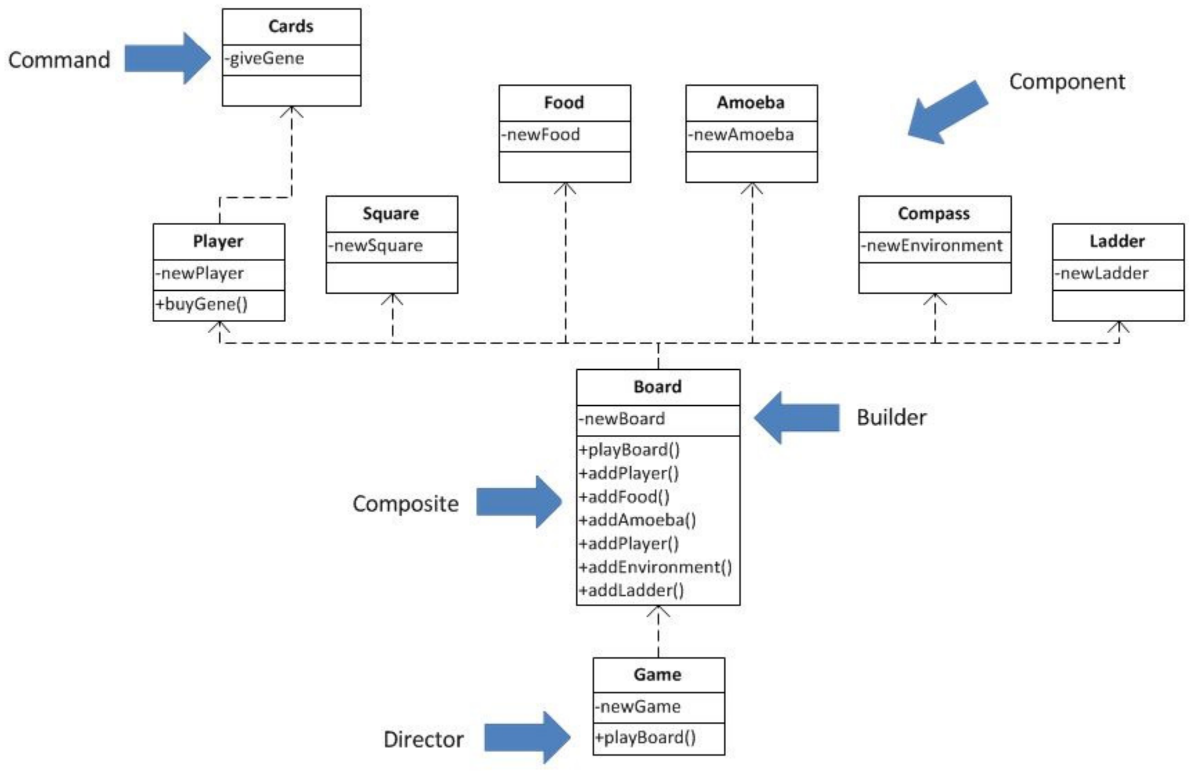
First we have an Object Oriented Design with objects like Squares, Players, Amoebas and the Board. We used some general reusable solution to make them work together well.

The game class serves as a director and tells the board, which is the builder, to create a new board.

The board class itself is a composite where the components are the players, amoebas, squares and food class.

The player class then asks the cards class to buy a gene and it knows whether that one is already sold to another player or if it is still available.

The following figure on the next page shows the design of PrimordialSoup at a glance explained with patterns.



The board is the central Class in this game. It is involved in two patterns, is used by the game and it uses the player, square, amoeba and the other components.

Conclusion

To conclude, let's make some general observations:

We were able to use some patterns in PrimordialSoup, but not that many, which means that it is roughly the same hard to make some changes as it is hard to use. Depending on the further development it will either tip over to the more pattern side, which will make it harder to apply changes but easier to use or the opposite case will happen.

Some of the patterns just arise without any plans or development purposes. They are nice to have and don’t bother in further progress.

Acknowledgement

Thanks to Kent Beck for his article about the Cook’s Tour in JUnit that served us as a template.

Special thanks to Patricia Schwab and Mirco Kocher for creating, reading and gentle corrections.