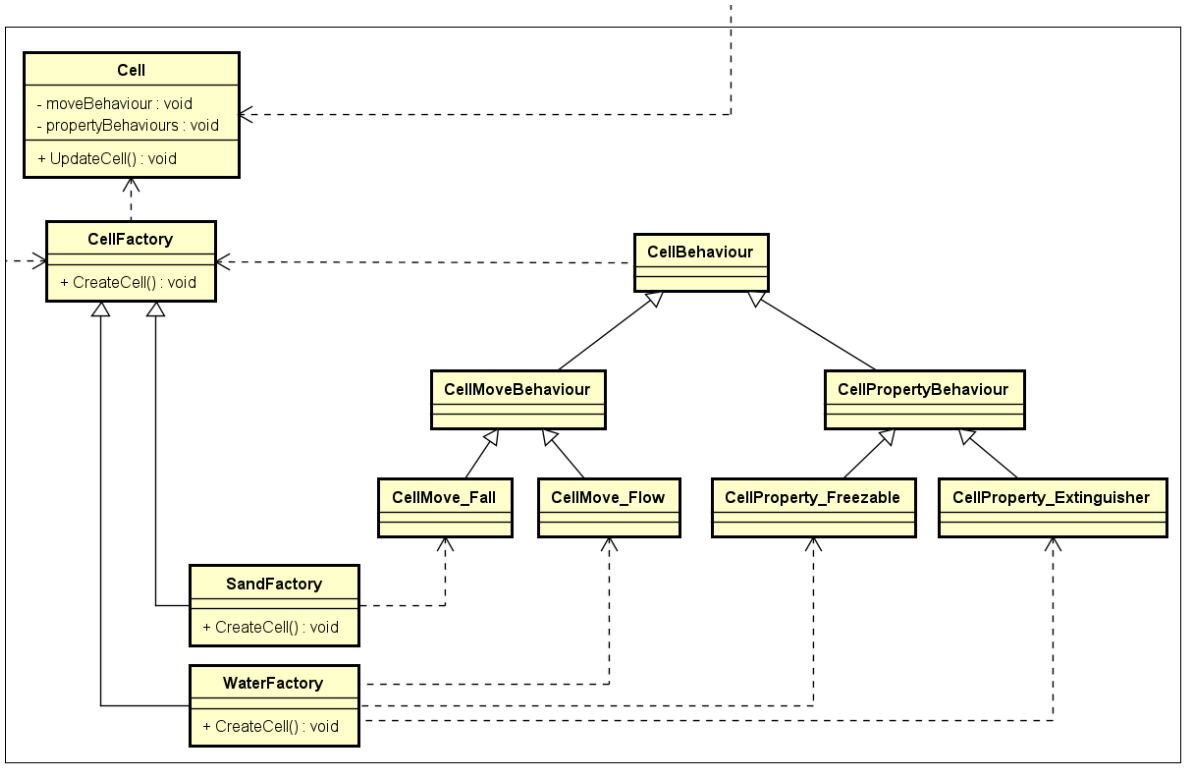
Final Project Report - Cellular Automation

Team member and responsibilities

- B05902042 林瑋毅: Game and software design, CellularMatrix-related classes, Report, Performance optimization
- B07902002 連崇安: Game and software design, Cell-related classes, Cells
- B09902044 林胤辰: Game and software design, Game-related classes, Game interfaces

The relations between the classes that you design

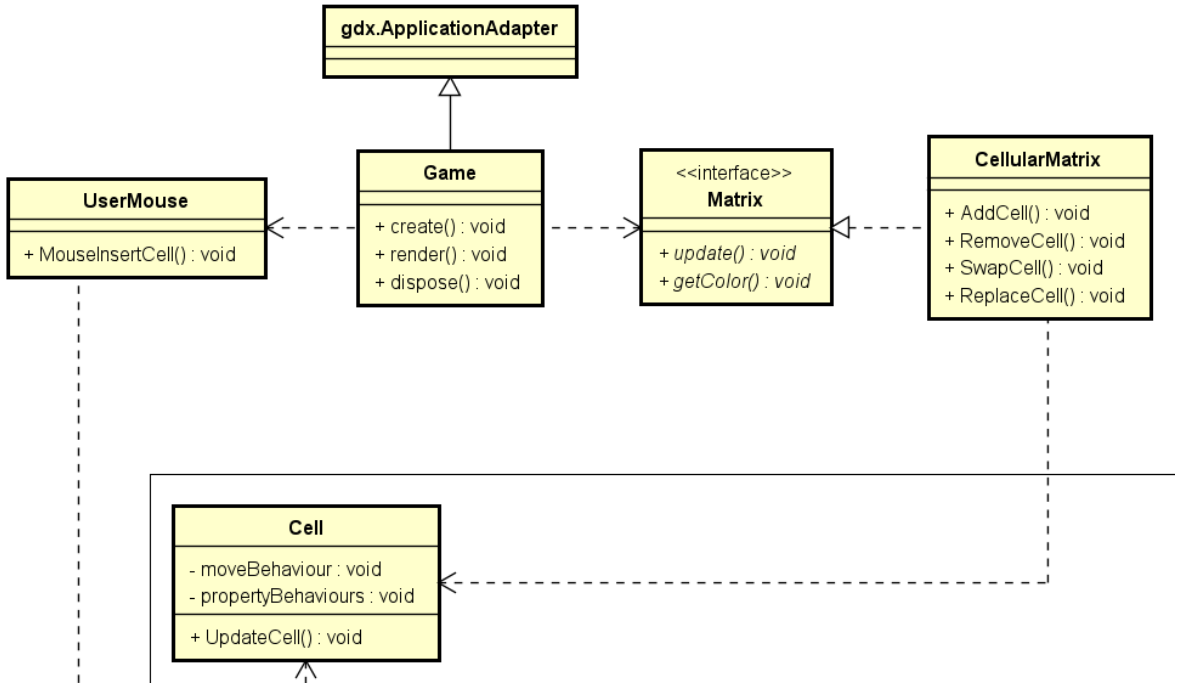


Cell-related classes

The above diagram shows a simplified relatinoship between cell-related classes. For convenience, selected kinds of cells are listed.

As we can see, `Cell` is the class that models the cells in our game. We can add a kind of move bevahior and several properties to the cells. An `UpdateCell` call to a cell update the cell based upon the bahaviours it contains.

`CellFactory` is the abstract class for creating cells. Different kinds of Factories depends on different behaviour classes based on domain knowledge. For instance, water moves like flow and is freezable and able to extinguish fire, so it dependes on `CellMove_Flow`, `CellProperty_Freezable`, and `CellProperty_Extinguisher`.



`Matrix.java` and `CellularMatrix.java`

The above diagram shows a relationship betwen `Matrix` and `CellularMatrix`. `Matrix` is an interface that allows users to get the color of each cell on the 2D map, and ask the map to update. `CellularMatrix` is a class that implements `Matrix` interface, and additionally provides some functions so that users can manipulate the cells on the maps.

`Game.java`

`Game` is a class that extends the `ApplicationAdapter` of libgdx. It handles the initialization, rendering of the game, and also the game loop.

`UserMouse.java`

`UserMouse` handles the cell creation/deletion when mouse is clicked.

The advantages of your design

The advantages of our design includes

1. We isolate the `Cell` and `CellBehaviour`, so we can easily create/delete a kind of cells without changing others. That is, the creation and deletion of a kind of cell (e.g., Sand, Water, etc) satisfy the open-closed principle.
2. The `CellMoveBehaviour` and `CellPropertyBehaviour` are also extendable, so we can create different moving mechanism (e.g., `CellMove_Ascend`) and property on the cells (e.g., `CellProperty_Flammable`). Note that the addition and deletion of bevaviours classes also satisfy open-closed principle.
3. The implementation of `CellularMatrix` could be replaced. For instance, we may implement a `CellularMatrix` that supports parallelizing the process of updating all the cells. This would not affect other components of the software.

The disadvantages of your design

The disadvantages of our design includes

1. The mouse control of the game was inside the `Game` class. If we want to change the game interface, the `Game` class have to be modified, making this component less preferable.

Other packages that you have used

In this project, we also used `libgdx`.

libgdx

The main responsibility for `libgdx` includes

1. Rendering the cellular matrix
2. Poviding an input interface (e.g., mouse clicks) that allows us to use mouse clicks to play the game.

How to play your 2D game

For Linux and Mac OS users, run `make` to start the game. For Windows users, run `make -f Makefile.win` to start the game.

Introduction

After enting the game, you'll see an empty map. In the upper left corner, there's an indicator for the real-time frame-per-second (FPS). In the upper right corner, there's an dashboard that tells you what type of cells will generate if you do a left-click.

Operations

1. Left-click: Generate new cells on the location of the cursor.
2. Right-click: Change the types of the generated cells. Note that if the type becomes "Null", that means you can delete the cell on the location of the cursor by left-clicking.

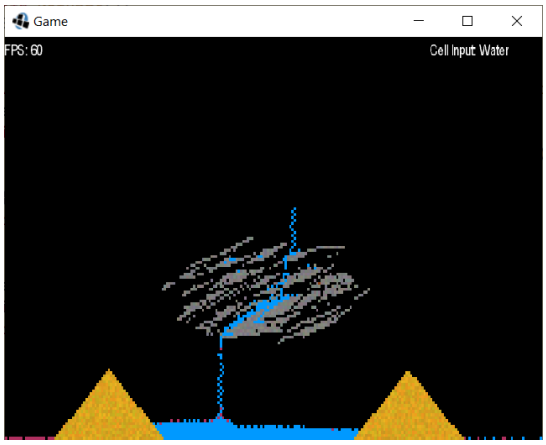
Supported types of cells

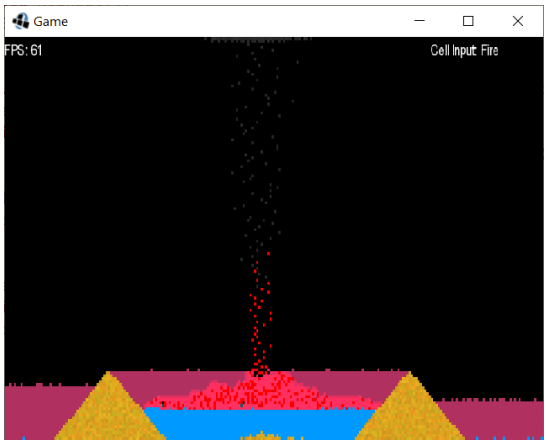
Currently, we suppot

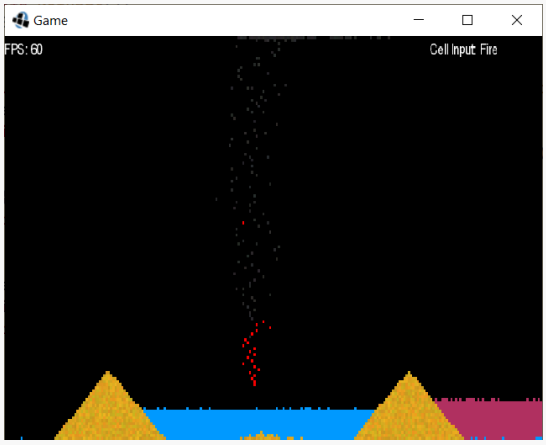
1. Fire
2. Ice
3. Oil
4. Rock
5. Sand
6. Smoke
7. Steam
8. Water
9. Wood

You can play with them and see how they interact. For isntance, if you put fires on oil, they burn!

Example images


Game
FPS: 60
Cell Input: Water


Game
FPS: 61
Cell Input: Fire


Game
FPS: 60
Cell Input: Fire