**Проект по курсу «Основы программирования Java»**

**Сыромятникова Алексея, группа КС-40**

Описание.

Проект представляет собой игру под названием «Жизнь» («Conway’s Game of Life»), которая имитирует клеточный автомат, придуманный английским математиком Джоном Конвеем в 1970 году.

Игровое поле реализовано с помощью класса JTable и имеет размерность 70х70 клеток. Каждая клетка представляет ячейку, которая может быть «жива» или «мертва» (соответствует белому или черному цвету клетки). В приложении имеются кнопки JButton «Start game», «Stop game», «One step», «Clean cells», а также переключатели JCheckBox «Unlimited borders» и «Drawing mode».

* Start game – запускает игру
* Stop game – останавливает игру
* One step – производит одну итерацию игры (одна перезапись всех ячеек и их перерисовка)
* Clean cells – очищает поле (устанавливает все ячейки в состояние «мертва»)
* Unlimited borders – включает режим «бесконечных границ». Превращает квадратное поле в эмуляцию поверхности тора посредством соединения левой границы с правой и верхней с нижней, но отображается все равно в виде квадрата (по умолчанию отключён).
* Drawing mode – режим позволяющий «рисовать» курсором мышки вместо точечного заполнения клеток (по умолчанию отключён).

В процессе игры все кнопки и переключатели блокируются, кроме кнопки «Stop game». Соответственно при окончании игры по условию или при искусственной остановке кнопки и переключатели разблокируются обратно, а кнопка «Stop game» наоборот блокируется.

Игра заканчивается по условию в тот момент, когда все ячейки мертвы или если на очередной итерации ячейки никак не изменились, то есть образовалась стабильная конфигурация.

Правила игры:

1. Если текущая ячейка жива и имеет две или три живые прилегающие ячейки, то на следующей итерации ячейка остаётся живой. В ином случае ячейка умирает на следующей итерации от переполнения или недостатка соседей.
2. Если текущая ячейка мертва и имеет три живые прилегающие ячейки, то эта ячейка «оживает» на следующей итерации, иначе ячейка остаётся мёртвой.
3. В процессе игры игрок является наблюдателем и не может влиять на её ход. Он может только задать начальную конфигурацию игрового поля и задать режим игры.

Структура и связь классов.

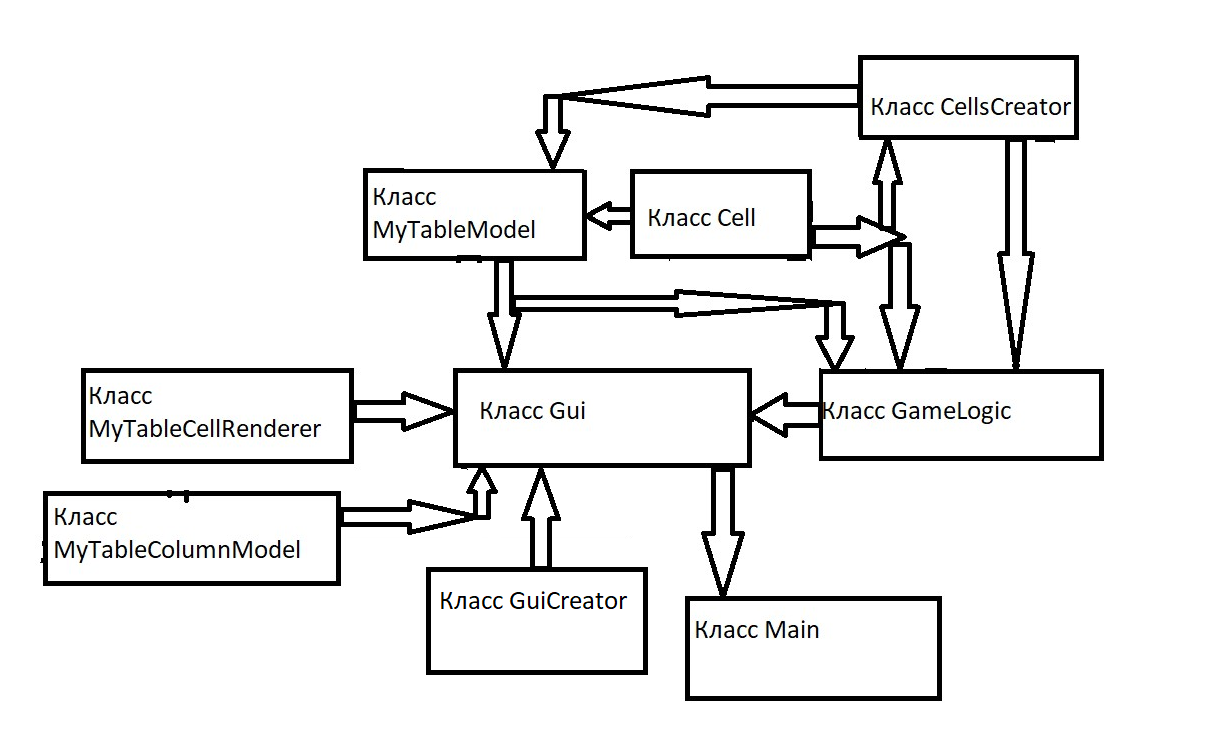


Рисунок 1. Связь классов.

* Класс Cell – имитирует ячейку
* CellsCreator – имеет статические методы, которые инициализируют матрицу ячеек типа Cell. Используется в GameLogic и MyTableModel.
* MyTableCellRenderer – является классом-рисовальщиком. Перерисовывает ячейки при их изменении в MyTableModel, в зависимости от их состояния.
* MyTableColumnModel – содержит модель столбцов игрового поля. Необходим для корректного отображение игровой сетки.
* MyTableModel – содержит в себе модель данных, которую использует игровое поле для отрисовки ячеек, имеет в себе двумерный массив ячеек типа Cell.
* Класс GameLogic – содержит логику игры, т.е. правила, алгоритмы и все данные и манипуляции с ними, необходимые для работы игры.
* Класс Gui – интерфейс игры. Содержит в себе модели для отображения сетки, рендер, логику игры, а также все необходимые кнопки и переключатели с обработкой событий.
* GuiCreator – инициализирует необходимые элементы интерфейса
* Main – инициализирует экземпляр класса Gui и все☺.

**Российский химико-технологический университет**

**имени Д.И.Менделеева**

**Факультет информационных технологий и управления**

**Отдел дополнительной профессиональной подготовки**

Название предмета: \_\_\_Программирование на языке Java\_\_\_\_\_\_

Преподаватели: \_\_Женса А. В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| № Задания | 1 | 2 | Оценка |
| Балл |  |  |  |

(подпись преподавателя)