Описание классов

Класс StartUI
Содержит метод main, в котором мы
— Создаём доску (Board)
— Заполняем её ячейками
— Создаем игровой контроллер (UIController) и передаем туда массив ячеек
— Вызыает метод игрового контроллера (UIController), который отрисовывает доску

Класс Board
Отвечает только за заполнение массива ячеек.
Так как сигнатура массива использует абстрактный класс Cell, мы можем помещать туда любые классы, реализующие его (NumberCell, EmptyCell, BombCell)

Абстрактный класс Cell
Описывает для ячейки:
— место в массиве (ряд и колонка)
— состояние (открыта, закрыта, отмечена флагом)
— её содержание (текстовая строка)
— поведение (сигнатура использует абстрактный класс CellActions, поэтому мы можем помещать туда любые классы, реализующие его (NumberActions, BombActions, EmptyActions))

Класс NumberCell
Содержит два конструктора: стандартный и тот, который используется при рассчете количества бомб вокруг числовой ячейки (отличается наличием параметра content). В обоих задается тип поведения для ячейки с бомбой.

Класс BombCell
Содержит стандартный конструктор, в котором задается тип поведения для ячейки с бомбой. Может использоваться для дальнейшего усложнения функциональности игры.

Класс EmptyCell
Содержит стандартный конструктор, в котором задается тип поведения для пустой ячейки. Может использоваться для дальнейшего усложнения функциональности игры.

Абстрактный класс CellActions
Задает реакцию ячейки на клик левой и правой кнопок мыши. Содержит дефолтную реализацию соотсветствующих методов. Открытие каждой ячейки уменьшает значение количества неоткрытых ячеек (хранится в классе UIController). После открытия каждой ячейки вызываем метод класса UIController, который проверяет не наступило ли условие для успешного завершения игры.

Класс NumberActions
Реализует CellActions. Использует дефолтную реализацию всех методов.

Класс BombActions
Реализует CellActions. Использует собственную реализаций методов:
— Правая кнопка мыши — Ставит или снимает флажок, при этом изменяя количество верно проставленных флажков, которое хранится в классе UIController

— Левая кнопка мыши — Приводит к окончанию игры, вызывает метод gameOver из класса UIController

Класс EmptyActions
Реализует CellActions. Использует дефолтную реализацию метода для правой кнопки мыши. Использует собственную реализацию метода левой кнопки мыши:
— Если ячейка закрыта, открывает её и все другие пустые ячейки вплоть до первых числовых, включая их.

Класс UIController
Основной класс игры.

При создании получает массив ячеек, на основании которого проводит отрисовку доски с абсолютным позиционированием всех ячеек.

Создаёт массив аналогичного размера, в который записываются ссылки на компоненты CellView. Они представляют из себя расширенную версию JPanel, в которой добавлены некоторые параметры.

Это позволяет связать объекты в коде и их графическое представление, так как для каждой ячейки есть ответный CellView, который можно найти по тем же координатам во втором массиве (и наоборот)

Также содержит методы, которые отвечают за:
— перерисовку объекта, на который кликнули
— проверку наступления условия победы (количество верно поставленных флагов равно количеству бомб на поле ИЛИ количество неоткрытых ячеек равно количеству бомб на поле)
— перерисовку доски и вывод сообщения при победе игрока
— перерисовку доски и вывод сообщения при проигрыше игрока
— вспомогательные методы для работы описанных выше

Класс CellView
Не представляет из себя ничего интересного.

Класс ClicksListener
Реализует интерфейс MouseListener. Намеренно был выбран интерфейс, а не адаптер, чтобы расширение функциональности требовало минимальных вмешательств в код. В данной версии реализован только метод mouseClicked, остальные оставлены пустыми.

Метод:
— определяет какой CellView был кликнут и какая ячейка ему соответствует
— определяет нажатую кнопку
- если левая — вызывается метод leftClickAction кликнутой ячейки
- если правая — вызывается метод rightClickAction кликнутой ячейки

Описание взаимодействия классов

Приложение начинает работать с класса StartUI. Его основная задача — создать объект класса Board, вызывать метод его заполнения ячейками, а затем создать UIController, передать в него данные Board и вызывать инициализацию интерфейса.

Создание Board
Создается двумерный массив объектов класса Cell. Его размер устанавливается в соответствии с заданным количеством рядов и колонок поля.

При создании ячеек сначала в случайном порядке расставляются бомбы. Затем, возле них расставляются цифры, которые рассчитываются автоматически. На последнем этапе создаются пустые ячейки. Они располагаются в тех элементах массива, в которые не записано никаких ячеек.

В одно из свойств каждой ячейки записывается объект класса, определяющего реакцию на нажатие кнопок мыши (для бомбы — BombActions, для цифры — NumberActions, для пустой — EmptyActions).

В таком виде основной массив класса Board и передаётся в UIController.

Работа UIController
Конструктор UIController сохраняет массив в одно из локальных свойств. Так как в StartUI после создания UIController сразу же идет вызов метода initUi, сразу же начинается отрисовка доски.

— Задаются основные параметры фрейма (Jframe)
— Для каждой ячейки создается и сохраняется на такой же позиции, но в другом массиве, соответствующий CellView. Его внешний вид задается при помощи иконки (ImageIcon). По умолчанию выводится иконка закрытой ячейки.
— На каждый CellView вешается слушатель событий мыши ClicksListener.

После этого окно приложения показывается и ожидает действий пользователя.

Действия пользователя обрабатываются при помощи классов, реализующих CellAction, записанных в виде одного из свойств ячейки. Слушателю нужно лишь определить какая кнопка мыши была нажата и вызвать соответствующий метод ячейки (leftClickAction или rightClickAction).

Каждый клик меняет свойство state у ячейки (необходимо для выбора правильных картинок — речь об этом далее).

Клик левой кнопки мыши работает для ячеек по-разному:
— по бомбе — вызывает метод gameOver класса UIController
— по цифре — открывает цифру
— по пустой — открывает её, а также все другие пустые ячейки в блоке до первых цифровых, включая их

После клика проверяется, не наступило ли условие для победы (в данном случае — не равно ли количество оставшихся на поле неоткрытых ячеек количеству расставленных бомб).

Клик правой кнопки мыши работает почти одинаково: если флага нет — ставим его, если флаг есть — снимаем его. Если выполняется правый клик по бомбе, мы меняем количество правильно поставленных флагов в методе UIController и после проверяем, не наступило ли условие для победы (не равно ли количество верно пространных флагов количеству расставленных бомб).

Количество неоткрытых ячеек и количество верно проставленных флагов хранится в UIController

Когда наступило условие победы — с каждой ячейки удаляется слушатель, что делает их неактивными, выводится сообщение о победе

Когда наступило условие проигрыша — все ячейки открываются, с них удаляются слушатели, что делает их неактивными

Подстановка правильных картинок

Перерисовка ячейки происходит при каждом клике пользователя на ячейку, для которого реализован какой-либо метод.

В методе обработки клика всегда используется две команды:
— сначала изменение статуса
— затем перерисовка

В папке resources лежат изображения:
— Для пустой ячейки — hidden.png;
— Для отмеченной флагом — flagged.png;
— Для открытой бомбы — openedB.png;
— Для открытой цифры — opened1, opened2 … opened8

Путь к иконке используется в методе перерисовки и составляется из трёх частей:
— путь к пакету с иконками
— для закрытой ячейки — ее статуса (hidden или flagged), для открытой ячейки из ее статуса + содержания.
— расширения файла (в игре — «.png»)

Это безопасно, так как поле содержания ячейки («content») автоматически устанавливается при ее создании и является закрытым для изменения классами, которые не входят в пакет cells (содержит абстрактный класс ячейки и его реализации).

Возможности расширения и изменения приложения

Как и было указано в задании, я пытался спроектировать приложение максимально гибким к изменению. Приведу примеры того, что можно легко изменить при минимальном вмешательстве в код.

— Создать новые типы ячеек — создаем соответствующие классы, наследуем их от Cell и, при необходимости, реализуем свое поведение

— Изменить изображения ячеек — просто меняем картинки в пакете resources

— Изменить поведение ячеек при нажатии на разные кнопки — переписать соответствующие методы в классах, реализующих CellActions или в нем самом

— Добавить другой тип ввода информации — повесить на CellView другие слушатели и сделать им минимальную реализацию