УДК 004.9

ИЗУЧЕНИЕ СПОСОБОВ ХРАНЕНИЯ ИНФОРМАЦИИ ПРИ РАЗРАБОТКЕ ИГР. РАЗРАБОТКА ИГРЫ «ЛАБИРИНТ НА УДАЧУ»

Семерня А.С., студент гр. ПИб-191, II курс
Научный руководитель: Киреева К.А., ассистент
Кузбасский государственный технический университет
имени Т.Ф. Горбачева
г. Кемерово

За последние несколько лет компьютерные технологии плотно внедрились в нашу жизнь. К примеру: будильник, помогающий проснуться в указанное время даже тогда, когда этого делать не хочется, календарь, напоминающий о важных событиях, вроде дня рождения родственников или о приближении дня получения зарплаты, приложение для составления списка и так далее. Эти продукты объединяет то, что если бы у них не было способа как-либо «запоминать» информацию, нужную пользователю, и «вспоминать» её при необходимости, то они были бы бесполезны. Будильник, не способный «вспомнить» когда разбудить пользователя и календарь, не знающий какое событие пройдёт через неделю, будет бесполезен. Отдельную категорию приложений, активно использующих их способность к запоминанию информации, составляют игры. Даже в самых простых проектах без системы уровней и способности переключения между игроками существует хотя бы таблица рекордов и, несомненно, чем сложнее проект, тем больше информации о нём нужно хранить. Встаёт вопрос, в каком виде хранить эту информацию?

Большинство игровых проектов используют для хранения информации о конкретных игровых состояниях бинарные файлы. Бинарные файлы специально проектируют для хранения необходимой информации, причем структура может быть разной, в зависимости от игры. Например, на рисунках 1-3 представлена структура хранения файлов сохранений в игре «Stygian: Reign of the Old Ones», от студии «Cultic Games». На данном примере отлично показано разбиение сохранений по локациям и дате посещения локации (рис. 1), в каждой папке хранится архив с названием сохранения (рис. 2) и в каждом архиве непосредственно сам файл сохранения с расширением «.save» (рис. 3).

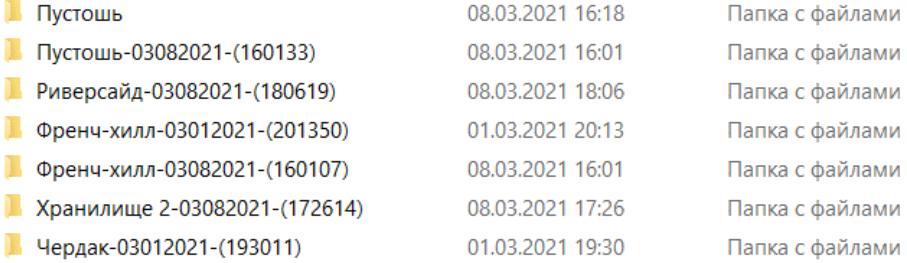

Рисунок 1 - Сохранения «Stygian: Reign of the Old Ones»

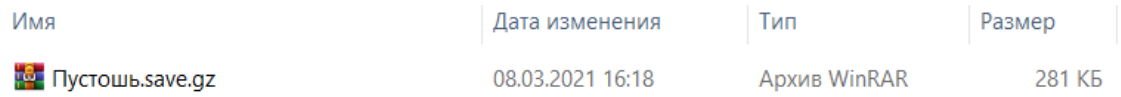

Рисунок 2 - Содержимое папки «Пустошь»

Рисунок 3 - Содержимое архива «Пустошь.save.gz»

На рисунках 4-5 приведен пример структуры файлов сохранения в игре «Stardew Valley», которая разработана Эриком Бароне и издана компанией «Chucklefish Games». Сохранения каждого игрока разбиты по папкам (рис. 4) и в каждой папке, хранятся бинарные файлы без расширения (рис. 5).

Рисунок 4 - Сохранения «Stardew Valley»

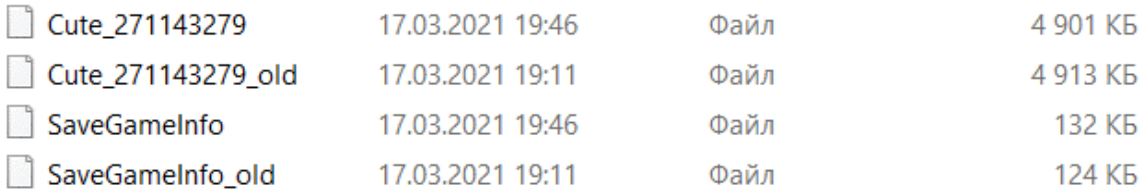

Рисунок 5 - Содержимое папки «Cute_271143279»

На рисунке 6 изображена структура хранения файлов сохранений игры «Spiritfarer», которая была разработана и выпущена компанией «Thunder Lotus Games». Файлы сохранений записываются последовательно и представляют собой бинарные файлы расширения «.sav».

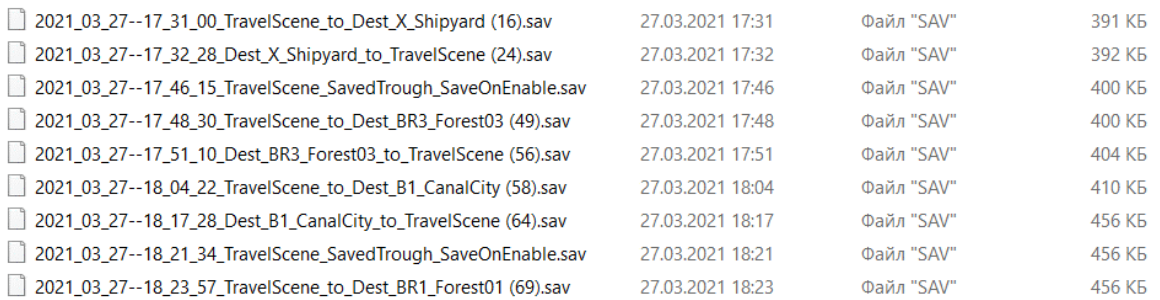

Рисунок 6 - Сохранения «Spiritfarer»

Представленные выше примеры демонстрируют, что структура хранения файлов изменяется от проекта к проекту, но практически всегда для этого используются бинарные файлы. Использование двоичных файлов, само по себе довольно удобно, но для их использования нужно прописывать дополнительную логику конвертирования данных в бинарный файл и обратно, помимо этого, не совсем удобно пользоваться двоичными файлами, когда дело доходит до отладки проекта и проверки его работоспособности в различных ситуациях. Приходится либо создавать сохранения различных отладочных со-

стояний и не запутаться в них, либо постоянно считывать данные одного состояния, менять их и запоминать снова. Данная работа является экспериментом по замене способа хранения информации в виде двоичных файлов на хранение всей необходимой информации в виде базы данных.

Для реализации игры были использованы следующие технологии:

1. C# - язык программирования.
2. Visual Studio - среда разработки.
3. MSSQL – средство управления базами данных.
4. Entity Framework - объектно-реляционный модуль сопоставления, позволяющий разработчикам .NET работать с реляционными данными с помощью объектов, относящихся к домену [1].

Базу данных условно можно разделить на 2 сегмента (рис. 7). Первый - три связанных таблицы «chests», «items», «monsters» и «effects» которые, соответственно, хранят информацию о сундуках, которые можно найти на уровне, предметах, которые можно найти, открывая сундуки, монстрах, которых так же можно встретить, обыскивая сундуки, и эффектах или «проклятиях» которые враги накладывают на игрока в случае проигрыша. Второй – две связанных таблицы «players», «lvls» в которых хранится информация, соответственно о всех игроках, игру за которых можно продолжить и о всех уровнях, которые когда-либо встречались в процессе игры.

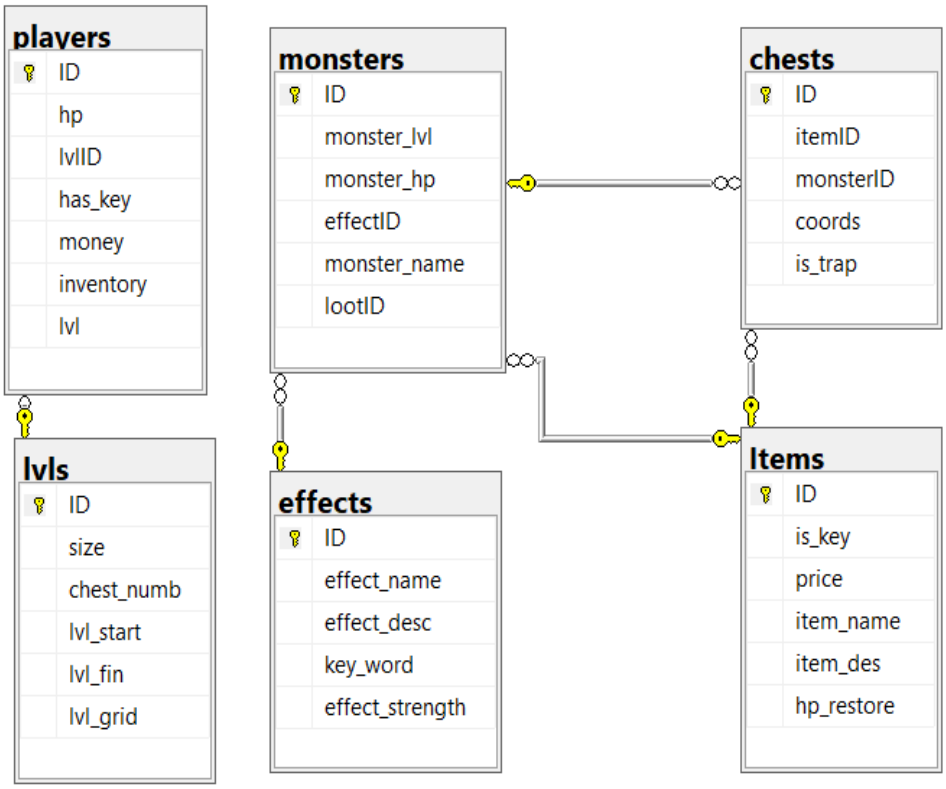

Рисунок 7 - Диаграмма используемой базы данных

Сама игра является представителем жанра «Roguelike». Главной особенностью жанра является никогда не похожие на предыдущие уровни. В данном случае, в качестве карты уровня генерируется квадратный лабиринт

заданной размерностью (не больше 61), в котором случайным образом расставляются так же случайно наполненные сундуки. Так как карта уровня представляет лабиринт, то было принято решение сократить поле видимости игрока до некоторой области вокруг себя для усложнения игрового процесса.

Разные элементы карты, видимые игроком на игровом поле обозначаются квадратами различных цветов (рис. 8): проходы или коридоры по которым может передвигаться игрок – белые, стены, ограничивающие перемещение – черные, сундуки – красные а сам игрок – зеленый. Переход на следующий уровень всегда находится в нижнем правом углу лабиринта и обозначается светло-оранжевым цветом.

У игрока есть возможность в любой момент времени использовать предмет, который есть у него в инвентаре. Управление реализованно с помощью компьютерной мыши, так и с помошью нажатия кнопок up, left, right и down, так и клавишами клавиатуры W, A, S и D.

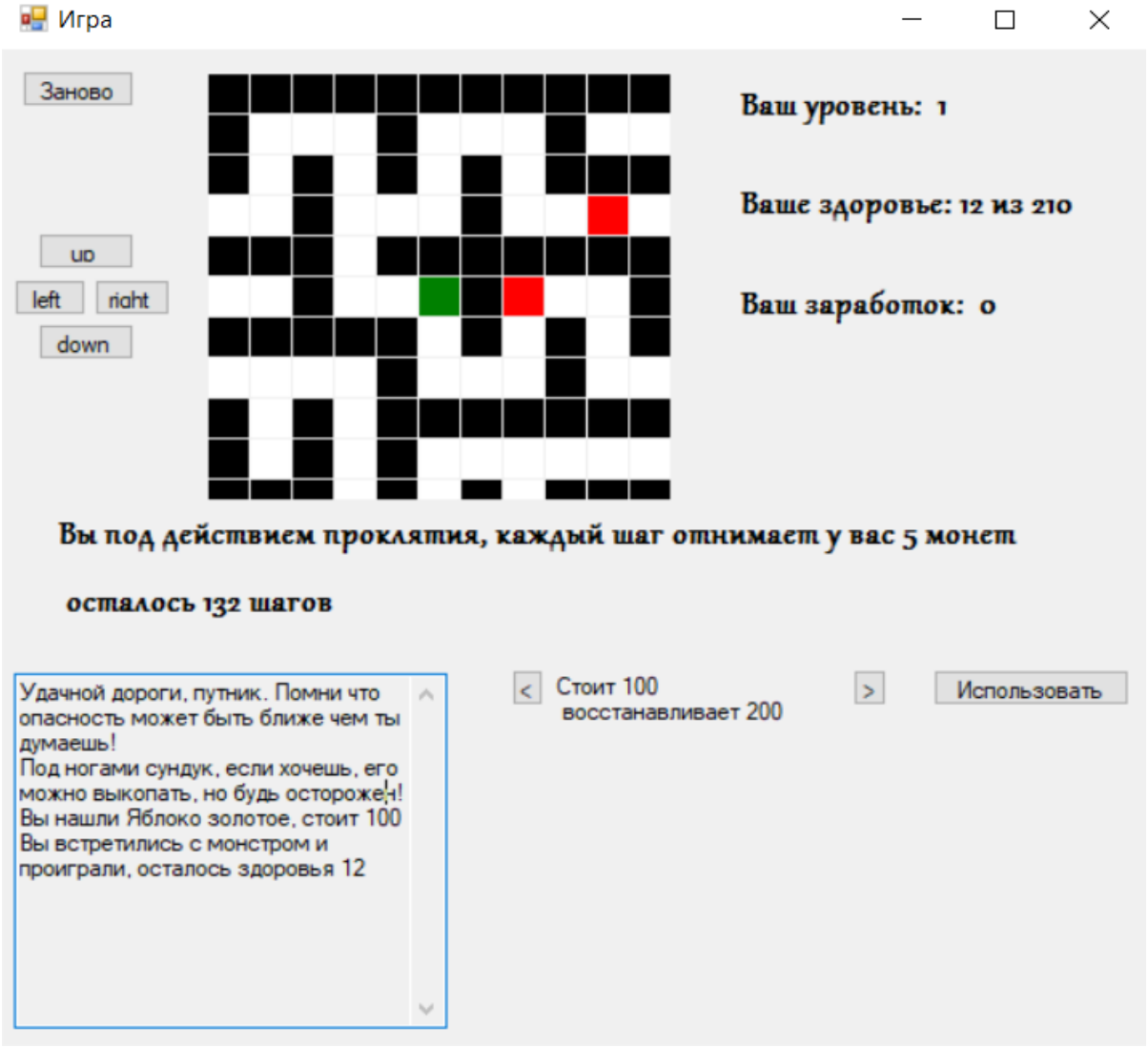

Рисунок 8 – Интерфейс главного окна разработанной игры

При запуске программы, первым открывается окно, которое предлагает пользователю либо начать новую игру, либо выбрать одну из сохраненных игр (рис. 9).

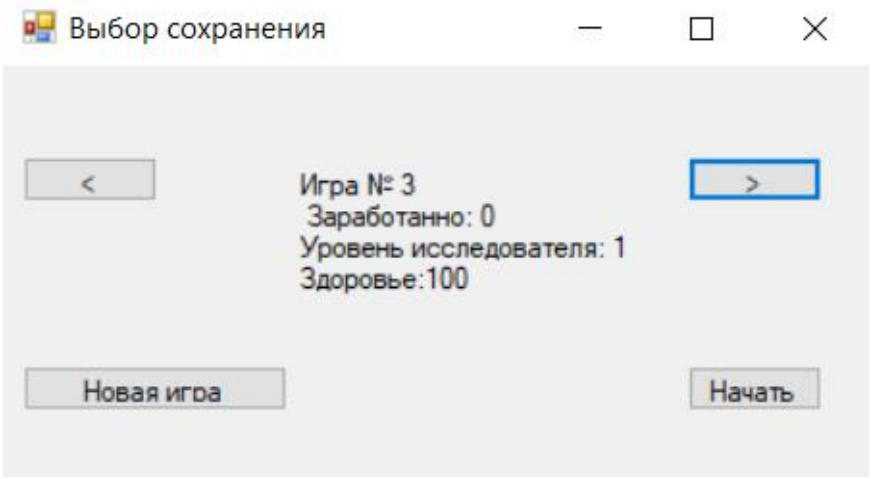

Рисунок 9 – Интерфейс начального окна с возможностью выбора сохранения

В итоге хранение информации об игре в форме базы данных на этапе разработки удобно, по той причине, что, к необходимой информации всегда есть доступ, с помощью средства управления базами данных и в случае необходимости информацию, которая была записана неправильно, можно исправить без лишних усилий. В то же время, использование Entity Framework позволяет больше сконцентрироваться на создании игровой логики и наполнении игры контентом, а не на способе связи базы данных и приложения.

Список литературы:

1. Сайт «Entity Framework | Microsoft Docs» [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/aspnet/entity-framework, свободный (дата обращения 25.03.2021).
2. Сайт «Control Класс(System.Windows.Forms) | Microsoft Docs» [Электронный ресурс]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.control?view=net-5.0, свободный (дата обращения 20.03.2021).
3. Сайт «Сайт о программировании METANIT» [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/entityframework/2.4.php, свободный (дата обращения 20.03.2021).