

Т. В. Земченко
(ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **Е. И. Сукач**, канд. техн. наук, доцент

ИСПОЛЬЗОВАНИЕ UNITY ДЛЯ РЕАЛИЗАЦИИ ИГРОВЫХ МЕХАНИК ИГРЫ «ТЕТРИС»

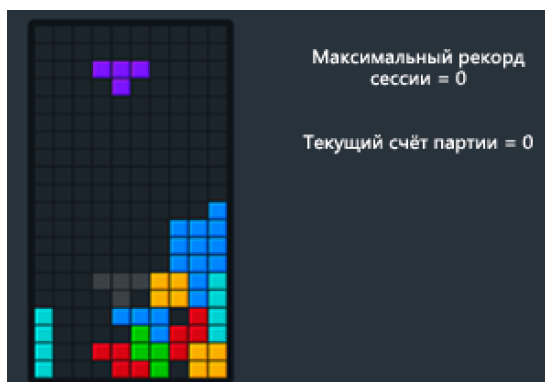
«Тетрис» – одна из самых популярных и узнаваемых классических компьютерных игр, которая уже не одно десятилетие остается в числе самых востребованных развлечений среди геймеров разных возрастов. Ее популярность обусловлена простыми, но увлекательными игровыми механиками, высокой динамичностью и возможностью совершенствовать свои навыки, развивая логическое мышление и скорость реакции. Благодаря своей универсальности, «Тетрис» представляет собой отличную платформу для изучения алгоритмов, освоения принципов работы игровых движков, а также применения объектно-ориентированного программирования. Разработка этой игры актуальна как для начинающих, желающих познакомиться с основами дизайна и программирования, так и для более опытных специалистов, стремящихся углубить свои знания в области структур данных, паттернов проектирования и оптимизации игровых механик. Этот процесс представляет собой комплексный подход к созданию интерактивных приложений, что делает его полезным для широкого круга специалистов.

В докладе рассматриваются аспекты программной реализации классической игры «Тетрис» с использованием игрового движка Unity. Освещаются ключевые игровые механики, особенности проектирования, структура кода, а также технические аспекты, обеспечивающие удобство управления и визуального восприятия игрового процесса. Разработка игры велась с использованием языка программирования C#. Применение игрового движка Unity позволило эффективно работать с визуальным интерфейсом, анимацией и графикой, а также обеспечило удобные средства для обработки данных и оптимизации игрового процесса.

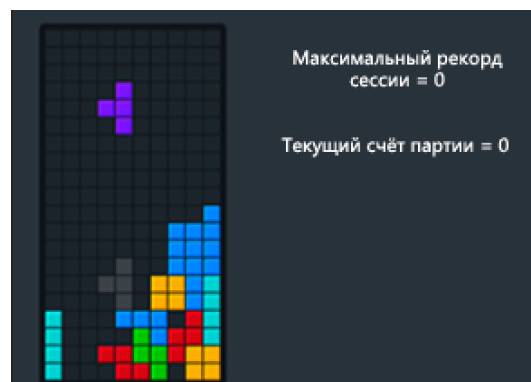
В ходе разработки использовались различные алгоритмические методы, способствующие точному и корректному воспроизведению всех игровых механик. В частности, были реализованы алгоритмы, отвечающие за падение фигур, их перемещение по игровому полю, вращение, проверку столкновений с границами поля и другими фигурами, а также удаление полностью заполненных строк. Пример использования поворота фигур представлен на примере фигуры "Т" (рисунок 1 (а,б)). Пример использования передвижения фигур представлен на примере фигуры "Т" (рисунок 1 (в,г)). Пример очистки поля и начисления очков (рисунок 1 (д,е)).

Для обеспечения корректной работы игровых механик была разработана специализированная структура данных, хранящая текущее состояние игрового поля и фигур, что позволило минимизировать вероятность возникновения ошибок и добиться высокой производительности. Отдельное внимание было уделено созданию интуитивно понятного пользовательского интерфейса, включающего в себя такие элементы, как счетчик набранных очков, а также отображение «тени» падающей фигуры, позволяющей игроку предугадывать место ее приземления и принимать более стратегические решения.

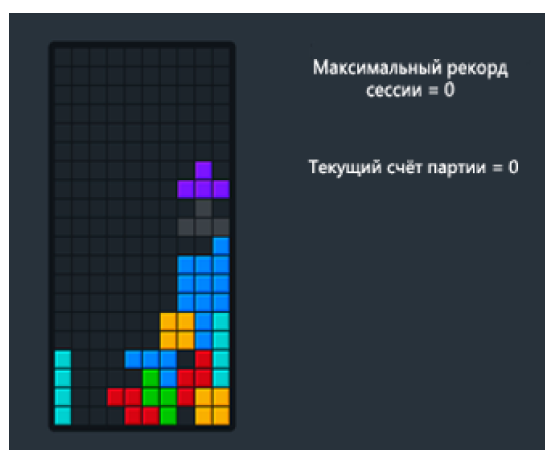
Применение паттернов проектирования, таких как MVC (Model-View-Controller), позволило создать структурированную и гибкую архитектуру приложения, обеспечивающую удобство в сопровождении, модификации и масштабировании проекта. Благодаря этому код игры стал более понятным и легко расширяемым, что делает возможным дальнейшее добавление новых функций, таких как усложненные уровни, различные режимы игры или дополнительные визуальные эффекты. В процессе оптимизации был обеспечен эффективный менеджмент памяти, что позволило добиться плавной работы игры даже на устройствах с ограниченными вычислительными ресурсами. Это стало возможным благодаря продуманной организации кода, минимизации избыточных вычислений и эффективному управлению объектами в игровом пространстве.



а



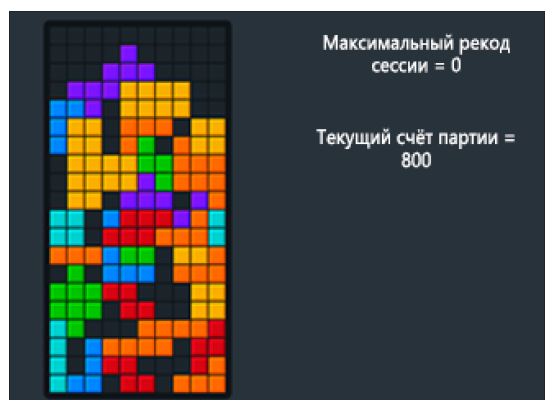
б



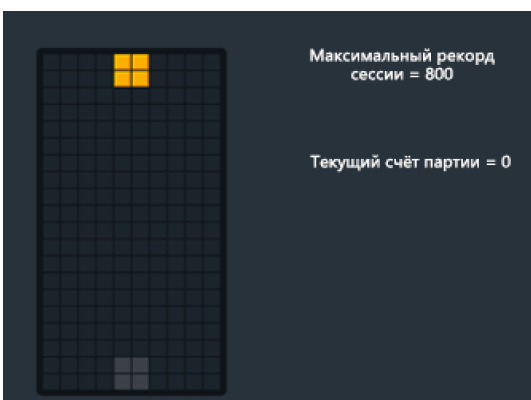
в



г



д



е

Рисунок 1 – Пример использования фигур: а – Фигура “Т” в начальном состоянии; б – Фигура “Т” повернута вправо; в – Передвижение фигуры “Т”; г – Передвижение фигуры “Т”; д – Ситуация перед проигрышем; е – Очистка поля

Разработанный алгоритм не только полностью воспроизводит все оригинальные правила игры «Тетрис», но и позволяет легко адаптировать проект для последующего изучения и модификации. Созданная архитектура дает возможность быстро вносить изменения, экспериментировать с различными параметрами и игровыми механиками, что делает данный проект отличной учебной платформой для начинающих программистов и разработчиков игр. Представленные методы и алгоритмы могут быть применены не только в «Тетрисе», но и в разработке других игр с аналогичными механиками, например,

головоломок или аркадных игр с падающими объектами. Это подтверждает универсальность полученных решений. Разработанные методы обработки игровых данных, алгоритмы проверки столкновений, а также система управления игровыми объектами могут быть адаптированы и использованы при создании других логических и аркадных игр, требующих динамического взаимодействия элементов и сложных алгоритмических решений. Например, данные алгоритмы могут быть применены в играх жанра «матч-3» для обработки совпадений и удаления элементов с поля, в платформерах для точного определения контактов персонажей с окружением, а также в стратегиях реального времени, где требуется динамическое управление юнитами и их взаимодействие с игровым миром.

Результаты исследования могут быть использованы для обучения разработке игр на платформе Unity и языке C#. Данный проект представляет собой не только практический пример создания классической игры, но и ценный образовательный материал, который может быть полезен как начинающим разработчикам, так и более опытным специалистам, желающим углубить свои знания в области программирования и работы с игровыми движками. На основе данного проекта могут быть разработаны специализированные учебные курсы, включающие в себя теоретические и практические занятия, направленные на изучение алгоритмов обработки данных, принципы построения архитектуры программных решений, а также освоение методов оптимизации производительности игровых приложений.

Разработанный проект может служить основой для создания учебных материалов и приложений, помогающих изучать программирование и алгоритмы на примере игры «Тетрис». Благодаря продуманной архитектуре, включающей модульную структуру кода и использование распространенных паттернов проектирования, таких как MVC (Model-View-Controller), студенты и разработчики смогут легко адаптировать код под собственные нужды. Они смогут экспериментировать с новыми механиками, такими как ускорение падения фигур, изменение формы блоков, а также добавление бонусных элементов или новых режимов игры. В рамках образовательных курсов данный проект может быть использован для демонстрации принципов работы с игровыми объектами, системами анимации, пользовательскими интерфейсами и взаимодействием игровых компонентов. Учебные материалы, основанные на этом проекте, могут включать как детальные руководства по его реализации, так и практические задания, направленные на закрепление полученных знаний.

Представленные подходы и алгоритмы могут быть успешно и эффективно применены в процессе создания других увлекательных и интересных игр с аналогичными игровыми механиками, что лишний раз подтверждает универсальность и широкую применимость полученных решений. Разработанные методы обработки игровых данных, продуманные алгоритмы проверки столкновений, а также гибкая и эффективная система управления игровыми объектами могут быть легко адаптированы и использованы при разработке других логических и аркадных игр, требующих динамического взаимодействия множества элементов и сложных алгоритмических решений, что значительно расширяет возможности разработки новых проектов. Таким образом, проект «Тетрис» представляет собой не только классический учебный пример, но и удобную отправную точку для создания новых оригинальных игровых приложений, что делает его действительно ценным ресурсом как для индивидуального обучения, так и для коллективной разработки в образовательных и профессиональных кругах, где важно учитывать множество факторов и подходов.