УДК 004.514

КОНЦЕПЦИЯ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ ДЛЯ ИГРОВОГО ПРИЛОЖЕНИЯ «LITTLE ESTATE»

Е. И. ЮРЧИШКО
(Представлено: канд. техн. наук, доц. В. М. ЧЕРТКОВ)

Описаны особенности игрового жанра симулятор фермы. Приводятся функциональная структура разработанного приложения. Описывается принятая концепция построения графического интерфейса пользователя. Представлены результаты разработки игрового приложения «Little estate» жанра симулятор фермы на игровом движке Unity.

Введение. Видеоигра (компьютерная игра), в широком смысле – любая игра, для функционирования которой необходима электронно-вычислительная машина [1].

Игры рассматриваются в основном как возможность проведения досуга, но при этом их роль не настолько однозначна. Многообразие жанров позволяет расширить возможности использования видеоигр. Игры-симуляторы ферм помогают развить стратегическое мышление игрока во время менеджмента ресурсов. Симуляция различных ситуаций из реальной жизни помогает безопасно пережить события, не выходя из дома.

Например, в 2008 году в массовой многопользовательской ролевой онлайн-игре «World of Warcraft» при стечении игровых обстоятельств возникла виртуальная эпидемия. Это происшествие не было запланировано разработчиками: «порченая кровь» – негативный эффект, отнимающий у персонажей очки здоровья, распространился между всеми персонажами и поломал игровой процесс. Ситуация привлекла интерес эпидемиологов, которые сопоставляли поведение игроков в онлайн-игре с поведением людей во время реальных эпидемий, рассматривая происшествие как полезную модель [2]. Таким образом, игры-симуляторы могут приносить определенную пользу в научной деятельности и для развития навыков игрока.

Разработка игр все чаще происходит с использование игровых движков, которые позволяют стандартизировать процесс разработки некоторых компонентов. Например, написание физики, работа с игровыми сценами и их компонентами. Для небольшого проекта с использованием 2D или простой 3D графики целесообразно использовать игровой движок Unity. Unity, на данный момент, одна из самых популярных сред разработки интерактивного 2D- и 3D-контента [5].

Unity предоставляет на выбор несколько лицензий, одна из которых является бесплатной для некоммерческих проектов. Unity – игровой движок, в котором логика игры и значительная часть анимации интерфейса создается посредством добавления скриптов на языке C#. Для их написания наиболее удобным инструментом является JetBrains Rider – кроссплатформенная интегрированная среда разработки программного обеспечения для платформы .NET, а также хорошо совместимая с Unity. Количество интеллектуальных механизмов для рефакторинга кода позволяют писать безошибочный код на языке программирования C# гораздо быстрее.

Постановка задачи и проектирование игрового приложения.

Функциональное назначение разрабатываемой игры состоит в развитии у игрока стратегического мышления благодаря менеджменту ресурсов при развитии фермы, а также тренировке художественного видения, так как игра представляет из себя инструмент для творчества.

К основным функциям игрового приложения относятся: лаконичный и функциональный пользовательский интерфейс в меню, на главной сцене с фермой, а также в других локациях; отображение ресурсов игрока (валюты); отображение характеристики игрового персонажа (выносливости); система покупки предметов для развития или декора фермы; пользовательские настройки; менеджер музыки и звуков; менеджер окон и страниц.

Важным условием положительного мнения пользователя об игровом продукте является максимальная оптимизация. Использование пулов является обязательным условием для этого. Повторяющиеся игровые объекты, используемые постоянно и в неограниченном количестве, будут негативно влиять на производительность. При многократном создании и удалении объекта тратится большое количество памяти. Пулы помогают решить эту проблему. Заранее создается достаточное количество объектов на сцене в отведенном месте. В первом кадре они, пока не будут использованы, находятся в состоянии отключенной видимости и не подключены к физике. В коде можно взять требующееся количество объектов из пула и вернуть их, когда необходимость исчезнет, а затем снова использовать в нужной ситуации. Таким способом, объекты не пересоздаются, а значит затраты памяти максимально снижены.

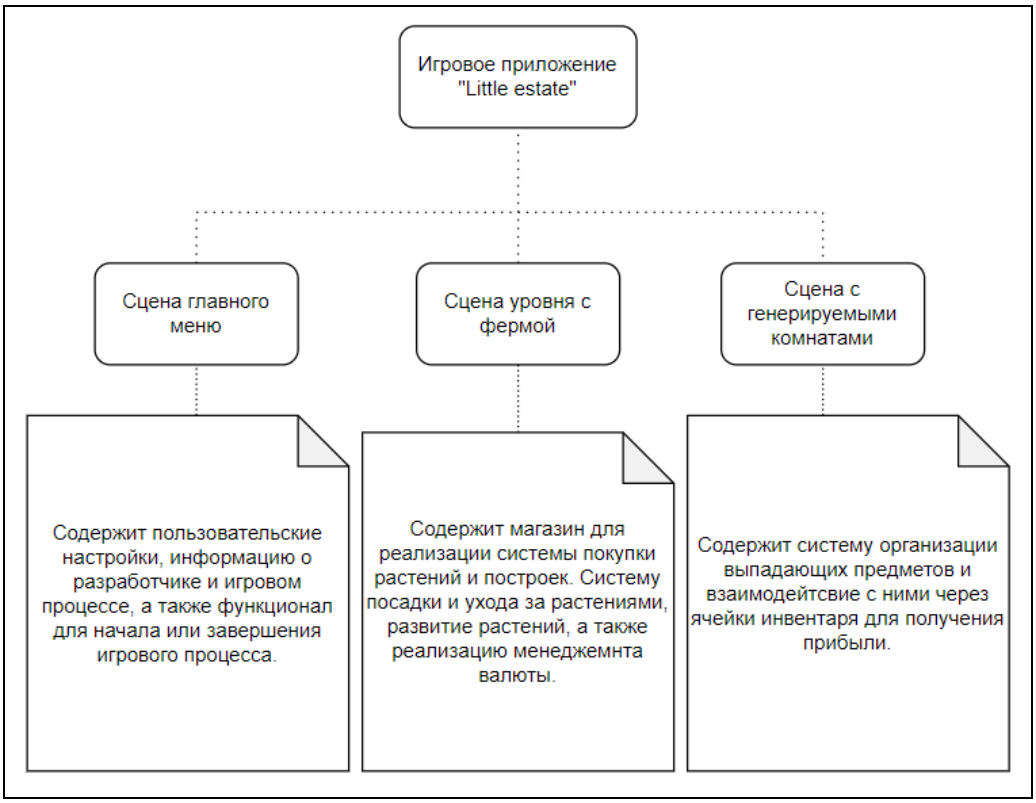

Рисунок 1. – Функциональная структура игрового приложения

«Little estate» – это симулятор фермы, который подразумевает уход за посаженными растениями, их рост, развитие, а также получение прибыли с выращенных экземпляров.

Растение в данной игре представляет из себя абстрактное понятия, в которое входят такие типы как деревья, а также плодовые растения, представленные в качестве ассортимента игрового магазина в разработанном приложении. Сам абстрактный класс Plant не может находиться на объекте спроектированной игровой сцены Unity как компонент, потому как абстрактные классы в Unity не обладают такой функциональностью. Таким образом, были созданы два класса-наследника: плодовое растение и дерево. Методы жизненного цикла объекта были ими унаследованы и позволили реализовать общую логику без дублирования кода. Эти скрипты поместились на игровой объект растения, который используется пулом для размещения на сцене в момент запуска игры.

Растение обладает фазами роста. Фаза роста также является абстрактным понятием, обладающим общими характеристиками. Это абстрактный класс типа ScriptableObject, от которого наследуются конкретные конфигурационные файлы ростка и выросшего растения. Следует отметить, что скриптовые объекты являются удобными контейнерами для хранения конфигураций, к которым будет обращаться множество объектов в Unity-играх.

Информация в скрипт растения поступает как раз в назначенную переменную для PlantConfig, и позволяет при «посадке» растения сразу определить нужное отображения для растения, его тип, установить начальную фазу роста и обрабатывать эти данные.

Растение на каждую стадию роста обладает определенным внешним видом. Прогрессия фаз происходит при динамической смене времени суток в игре, а также при соблюдении условий ухода за растением. Чтобы новый спрайт растения при замене не «провалился» в текстуру поверхности земли, в методе роста присутствует блок кода, отвечающий за расчёт точки соприкосновения растения с полом. Для этого из объекта по вектору Vector3.down (0; -1; 0;), бросается специальный луч Raycast, который находит верхнюю границу коллайдера земли, и на найденную координату поверхности устанавливает объект таким образом, чтобы самая нижняя точка растения соприкасалась с ней. При смене размера спрайта у растения также должны изменяться границы коллайдера, чтобы столкновение с объектом работали корректно.

Стадии растения реализуют паттерн Состояние. Состояние – шаблон проектирования, который позволяет объекту изменять свое поведение в зависимости от внутреннего состояния [4]. Этот шаблон позволяет объекту изменять свое поведение при изменении внутреннего состояния, создавая впечатление, что объект изменяет свой класс. Это достигается путем инкапсуляции состояний в отдельные классы

и делегирования действий объектам этих классов. Базовый класс состояния в данном проекте: GrowthStage. Это абстрактный базовый класс, который определяет интерфейс для всех конкретных состояний.

Таким образом, растение является универсальным объектом, динамически изменяющим свои параметры. Такой подход позволяет избежать создания огромного количества заранее заготовленных экземпляров и сохранить ресурсы памяти. В пул объектов достаточно поместить один общий объект для растения, который при получении конфигурационного файла будет автоматически перенастраиваться нужным образом.

Рисунок 2. – Пример фаз роста растений

Одним из интерактивных объектов на сцене является магазин, при взаимодействии с которым открывается окно с товарами, разделенными на несколько категорий. Каждый доступный товар представлен в карточке, содержащей название, описание, цену и отображение предмета.

Доступное количество валюты у игрока отображается в левом верхнем углу экрана. Если цена товара не превышает доступное количество валюты игрока, то совершается покупка. Для покупки товара пользователь должен нажать левой клавишей мыши по кнопке «Shop» в выбранном товаре.

После успешной покупки у игрока списывается сумма за товар, окно магазина закрывается, а на сцене появляется купленный объект. Управление от передвижения персонажа передается к системе размещения объекта, то есть происходит переназначение клавиш.

Размещаемый объект можно установить только в свободном месте на сцене. Возможность размещения отображается подсветкой зеленого цвета. Если объект разместить в выбранном месте запрещается, то отображение объекта становится красным. Если игрок хочет отменить покупку, то нужно нажать клавишу «ESC», после чего отображение предмета пропадает, происходит начисление валюты в полном размере, а управление переходит обратно к персонажу. Для размещения объекта следует нажать левую клавишу мыши.

Одним из основных товаров в магазине являются растения. Каждое растение имеет несколько фаз роста, а также систему «жажды». После посадки игрок может взаимодействовать игровым персонажем с объектом «дом», после чего запустится анимация смены внутриигровых суток, а каждое посаженное растение перейдет к следующей стадии роста. Также с началом каждого нового дня все растения требуют полива. Цвет растения с активной системой жажды изменяется с обычного на коричневый.

При взаимодействии со взрослым предварительно политым растением, открывается интерактивное меню с опциями для продажи растения или получения с него урожая в виде саженцев.

Для переключения между пунктами меню также происходит переназначение клавиш: чтобы выбирать между пунктами меню, игроку нужно нажимать клавиши «W» и «S». Выбранный пункт подсвечивается зеленым. Для подтверждения действия пользователь нажимает «E». Пример меню взаимодействия с растением представлен на рисунке ниже.

Если игрок нажмет на клавишу «Q», то произойдет взаимодействие с предметом в активной ячейке инвентаря. Если предмет является ростком растения, то появится возможность разместить растение на участке, аналогичная с покупкой.

Рисунок 3. – Окно внутриигрового магазина

Рисунок 4. – Меню взаимодействия с растением

Заключение. При разработке концепции графического интерфейса пользователя, а также при его проектировании на всех сценах игры, были соблюдены все требования. Анализ аналогичных программных продуктов на рынке позволил принять обоснованные решения относительно концепции и направления игры, избежать непопулярных и неэффективных решений в конкурирующих играх.

Программный продукт успешно прошел все этапы тестирования без выявленных дефектов. Игра «Little estate» показала себя, как готовый к эксплуатации программный продукт, стабильный в использовании и с интуитивно понятным интерфейсом.

ЛИТЕРАТУРА

1. Подвальный, М. А. Видеоигра [Электронный ресурс] // Большая российская энциклопедия. – Режим доступа: https://bigenc.ru/c/videoigra-915272 – Дата доступа: 14.05.2024.
2. Olivetti, J. The Game Archaeologist: World of Warcraft's Corrupted Blood pandemic [Электронный ресурс] – Режим доступа: https://massivelyop.com/2018/03/17/the-game-archaeologist-world-of-warcrafts-corrupted-blood-pandemic/ – Дата доступа: 14.05.2024;
3. Максимов, Д. Движок Unity [Электронный ресурс] – Режим доступа: https://skysmart.ru/articles/programming/dvizhok-unity. Дата доступа: 10.04.2024 – 14.04.2024;
4. Попов, Е. Состояние (State) [Электронный ресурс] – Режим доступа: https://metanit.com/sharp/patterns/3.6.php. Дата доступа: 10.05.2024.