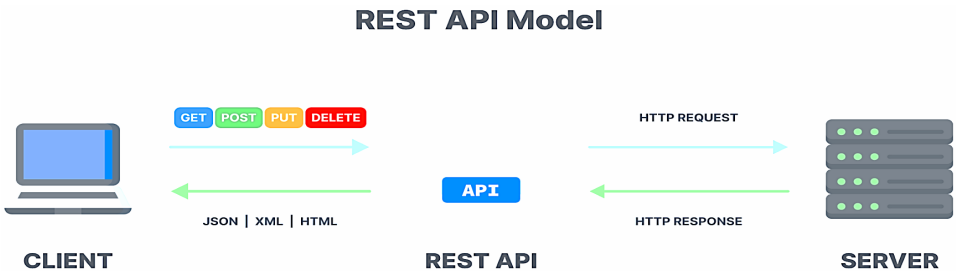

Рис. 2. Схема работы RESTAPI

Заключение. Таким образом, внедрение в учебную среду проектируемого *web*-приложения обеспечит эффективный учет и анализ учебной деятельности студентов, позволит автоматизировать процесс формирования отчетов и аналитики по результатам обучения, предоставит возможность упрощенного доступа к работе с необходимыми документами преподавателям, специалистам деканатов и администрации университета, сопровождающих учебный процесс.

Литература

1. MongoDB /«HowtoUseMERNStack: ACompleteGuide». – 2024. – Режим доступа: https://www.mongodb.com/languages/mern-stack-tutorial. – Дата доступа: 12.03.2024.
2. REST Framework / «Создание RESTful API с помощью Express и Node.js в стеке MERN». – 2024. – Режим доступа: https://medium.com/@ibrahimhz/building-a-restful-api-with-express-and-node-js-in-mern-stack-e3714f94d24b – Дата доступа: 12.03.2024.

РАЗРАБОТКА СЮЖЕТНОГО ИГРОВОГО ПРИЛОЖЕНИЯ В ЖАНРЕ «КОНЕЧНЫЙ РАННЕР» С ИСПОЛЬЗОВАНИЕМ АВТОРСКОЙ ГРАФИКИ В СРЕДЕ UNITY

О. В. Овчинина

Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», Республика Беларусь

Научный руководитель В. С. Захаренко

Рассмотрены основные проблемы при разработке игрового приложения «Echo-10» и их решения, такие как выбор средств разработки, архитектуры, процесс создания авторской графики, механик игры и сюжета.

Ключевые слова: игровое приложение, архитектура, графика, сюжет, разработка.

Игры в жанре «Раннер» давно укрепились в мире видеоигр. Они характеризуются процедурной генерацией уровней, созданием препятствий и системой подсчета очков. Под жанр «Конечный раннер» включает в себя наличие нескольких уровней и завершающий этап игры, что обеспечивает более детальное развитие сюжета игры.

Средства разработки. Для разработки игры был выбран язык программирования *C#*, игровой движок *Unity*, интегрированная среда разработки *Rider*, графический редактор *AdobePhotoshop* для создания оригинальной графики и сервис по управлению проектами *Notion*. Благодаря кросс сплатформенной среде разработки *Unity* приложение адаптируется как для персонального компьютера, так и для мобильного устройства.

Авторская графика. С помощью инструмента *AdobePhotoshop* была создана авторская графика для игрового приложения, с целью придания уникального и привлекательного визуального стиля, а также для создания неповторимой атмосферы в игре. Кат-сцена игры, в качестве примера, представлена на рис. 1.

Рис. 1. Кат-сцена игры

Архитектура. Архитектурные проблемы в играх, разработанных с использованием *Unit* – среды разработки компьютерных игр, включают в себя сложности с контролем состояний игровых объектов, а также зависимость от компонентов *Unity*, что может затруднить переносимость и интеграцию проекта. Неэффективное использование ресурсов и недостаточное разделение обязанностей также могут привести к проблемам производительности и сложностям в модификации проекта.

Чтобы решить эти проблемы была разработана собственная архитектура для приложения «*Echo*-10». Архитектурным решением является добавление компонентов на игровые объекты на стадии инициализации сцены.

Архитектура приложения состоит из трех модулей: *Libs*, *Infrastructure*, *Scene*. Модули в свою очередь разбиты на подмодули.

Модуль *Infrastructure* содержит основные элементы управления игрой.

Схема модуля *Infrastructure* представлена на рис. 2.

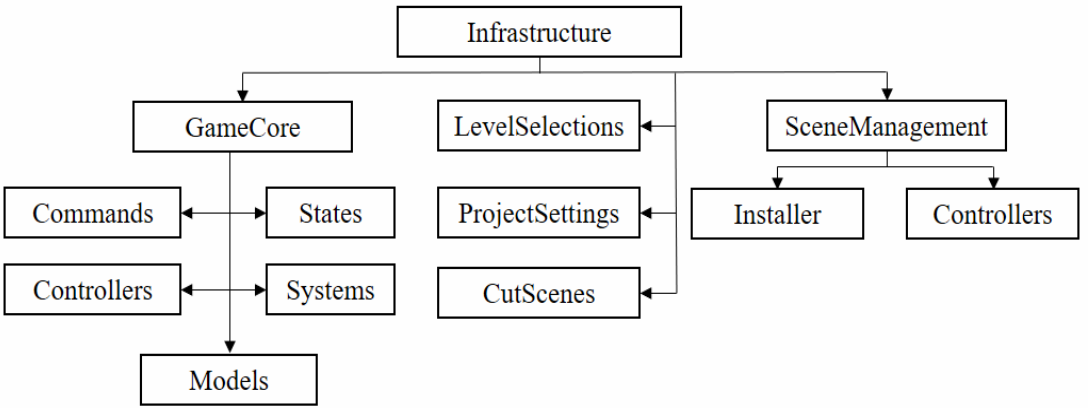

Рис. 2. Схема модуля *Infrastructure*

Подмодуль *GameCore* отвечает за основную логику игры. Он состоит из:

– шаблона проектирования «Команда» *Commands*– используется для инкапсуляции запроса в виде объектаи позволяет откладывать выполнение операции, добавлять ее в очередь, а также выполнять отмену операции;

– контроллеров *Controllers* – управляют состоянием игры;

– моделей *Models* – управляют состоянием объектов;

– состояний *States* – описывают поведение состояния игры;

– систем *Systems* –компоненты, ответственные за выполнение конкретной функциональности в приложении.

Подмодуль *SceneManager* включает в себя *Controllers* для обеспечения начальной настройки навигатора сцен и *Installer* для настройки зависимостей и установки компонентов.

Подмодуль *LevelSelections* включает в себя классы для смены уровня.

Подмодуль *ProjectSettings* отвечает за настройку проекта.

Подмодуль *CutScenes* включает в себя классы для смены кат-сцены.

Модуль *Libs* содержит функциональность библиотеки классов и предоставляет базовые инструменты для разработки игры. Схема модуля *Libs* представлена на рис. 3.

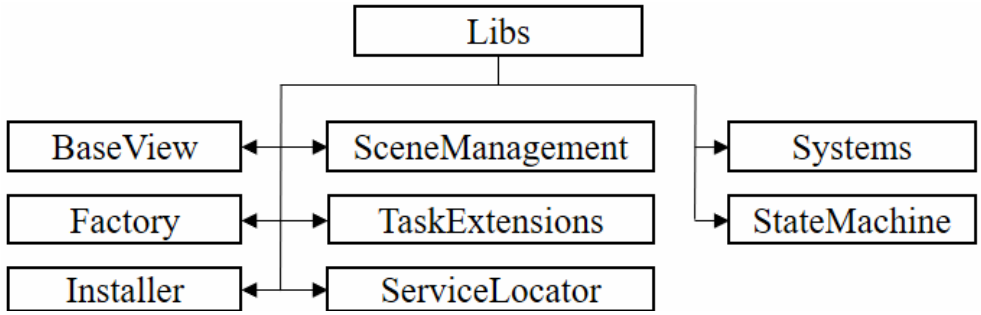

Рис. 3. Схема модуля *Libs*

Подмодуль *BaseView* отвечает за основной компонент архитектуры – класс с *MonoBehaviour*.

Подмодуль *Factory* содержит класс для создания префабов.

Инсталлеры позволяют структурировать и управлять зависимостями в приложении, облегчая его настройку и расширение, а подмодуль *Installer*представляет собой базовый класс для инсталлеров.

Подмодуль *ServiceLocator* реализует шаблон проектирования «Сервисный локатор», а его класс представляет собой контейнер служб, который предназначен для регистрации, хранения и получения различных служб или зависимостей в приложении.

Подмодуль *SceneManagement* обеспечивает гибкую систему навигации по сценам в *Unity*, позволяя легко добавлять новые сцены и настраивать их конфигурацию через объект *ConfigScenes*.

Подмодуль *StateMachine* реализует машину состояний, которая определяет все возможные состояния игры, их изменения в ответ на события или условия, а также действия, выполняемые в каждом из состояний.

Подмодуль *Systems* включает в себя группу систем.

Подмодуль *TaskExtensions* позволяет не прерывать игровой процесс в случае возникновения исключения.

Модуль *Scene* содержит реализацию функционала сцены. В данной архитектуре каждая сцена рассматривается как отдельный проект. В зависимости от функционала сцена разбивается на подмодули. Схема примера разбиения модуля *Scene* на подмодули представлена на рис. 4.

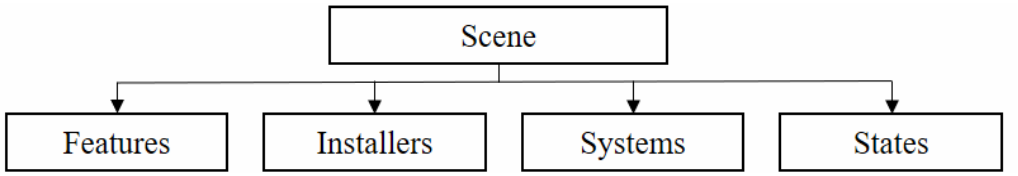

Рис. 4. Схема модуля *Scene*

Подмодуль *Features* отвечает за классы конкретных объектов, подмодуль *Installers* отвечает за первоначальную настройку параметров, подмодуль *States* отвечает за логику переключения между состояниями, подмодуль *Systems* реализует системы: перемещение игрока, столкновение объектов, анимация перехода между сценами и т.д.

Сюжет. В приложении «*Echo*-10» игрок управляет Мигелем – летучей мышью, лишенной эхолокации и вынужденной полагаться на новые технологии для ориентации. Сюжет раскрывается по ходу игры в виде небольших кат-сцен перед прохождением уровня.

Механики. Игра состоит из нескольких уровней, где игрок должен избегать препятствий и добраться до конца. Препятствия не видны на экране до тех пор, пока игрок не использует эхолокационный заряд. При этом они на короткое время подсвечиваются, и игрок должен запомнить их расположение. Количество зарядов на уровень ограничено до 10.

WEB-ПРИЛОЖЕНИЕ ДЛЯ АВТОМАТИЗАЦИИ РАБОТЫ ХОРЕОГРАФИЧЕСКОЙ ШКОЛЫ

В. А. Шевкунова

Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», Республика Беларусь

Научный руководитель Т. Л. Романькова

Описаны компоненты, необходимые для создания web-приложения по автоматизации работы школы: функционал, архитектура, схема базы данных, а также технологии программного комплекса.

Ключевые слова: автоматизация, *web*-приложение, трехслойная архитектура, SignalR.

В настоящее время современные предприятия и организации постоянно стремятся к повышению эффективности и производительности. Один из наиболее эффективных методов достижения является автоматизация процессов. Хореографическая школа – не исключение.

Автоматизация школы может значительно упростить и ускорить процессы работы. Так, использование специального программного обеспечения для управления расписанием занятий и регистрации на них позволит клиентам самостоятельно выбирать удобное время и день для занятий, а также оплачивать обучение онлайн.