УДК 004

Создание контролер управление персонажа на игровом движке Godot

Черкашин Александр Михайлович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан процесс создание персонажа для управления от первого лица на игровом движке Godot. В работе описан конструктор игр для создания персонажа от первого лица для заготовки. В результате работы был создан персонаж и использован конструктор игр Godot.
Ключевые слова: Godot, Character, GDScript.

Creating a character controller using the Godot game engine

Cherkashin Alexander Mihailovich
Sholom-Aleichem Priamursky State University
student

Abstract
This article describes the process of creating a character for first-person control on the Godot game engine. A game designer is in the works to create a first-person character for a template. As a result of the work, a character was created and the Godot game designer was used.
Keywords: Godot, Character, GDScript.

1 Введение

1.1. Актуальность исследования

Актуальность исследование заключается в том, что поскольку обеспечивает разработчиков мощным инструментарием для реализации увлекательного геймплея. Godot предоставляет гибкие средства для анимации, управления персонажем, а также возможность легкой интеграции с другими элементами игры. Создание персонажа в Godot позволяет разработчикам легко воплощать свои идеи в жизнь, создавая уникальный игровой опыт для игроков.

1.2. Цель исследования

Целью работы создания заготовки игрока для управления персонажа игровой движок Godot.

1.3. Обзор исследований

Исследование, проведенное Т. Салмела в бакалаврской диссертации, изучает инструмент разработки видеоигр с открытым исходным кодом Godot Engine. Популярность и жизнеспособность независимой разработки игр растут

вместе с появлением множества программных инструментов, доступных сегодня разработчикам, в данном работе автора исследователя была посвящена потенциалу Godot Engine для разработки игр, его дизайну, функциям и рабочим процессам [1].

Исследование, проведенное Ц. Андерссон, Т. Мелландер рассматривает возможность системы перемотки времени в игре позволит игроку вернуться в игровое время, чтобы переиграть определенный уровень или сеанс. В дипломной работе исследуется возможность реализации обобщенной системы перемотки времени, которая будет включена в движок Godot [2].

Х. Мэкелэ. в современных условиях все больше разработчиков видеоигр прибегают к использованию внешних игровых движков, поскольку они способствуют экономии ресурсов компании. Принятие решения о выборе подходящего движка для конкретного проекта представляет собой сложную задачу, зависящую от различных факторов, таких как масштаб проекта и желаемая визуальная точность графики. Одним из новейших игровых движков, который все больше разработчиков предпочитают, является Godot Engine. Тем не менее, его статус и возможности, особенно в области трехмерной разработки, остаются относительно недостаточно исследованными по сравнению с более распространенными движками, такими как Unity. В связи с этим возникла необходимость определить целесообразность применения Godot Engine при разработке трехмерных игр. Целью данного исследования стало проведение глубокого анализа использования этого движка в процессе создания трехмерной японской игры риичи-маджонг. Маджонг представляет собой существующую настольную игру, аналогичную покеру, но более сложную и более подходящую для трехмерной реализации [3].

2. Рабочий процесс

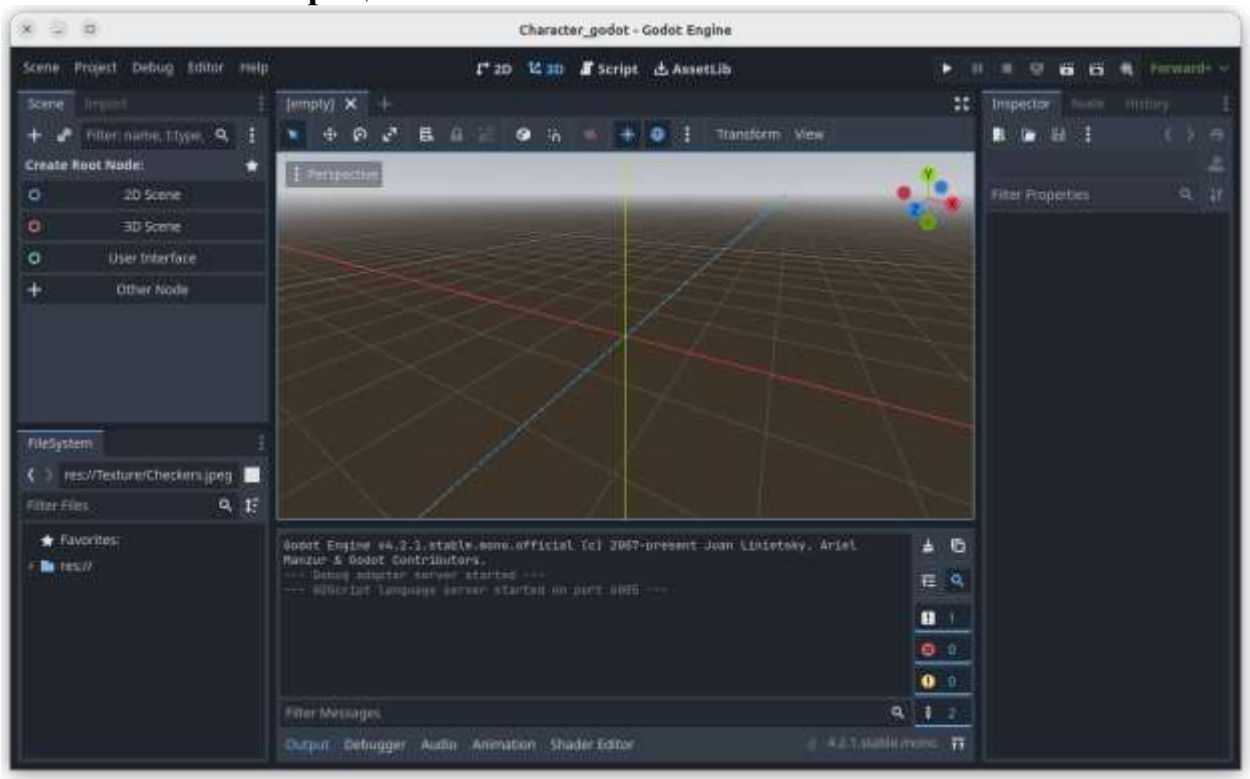

Рисунок 1. Интерфейс конструктор игр Godot

Мы создали проект и назвали «Character_godot» а затем создали 3D сцену и сохранили в файл «res://Level/Main.tscn» в сцене прилепили скрипт и назвали его «res://Level/Main.gd».

«Main.gd» выполняет перехват курсора для управление персонажа от третьего лица и ожидает ввод клавиатуры «ESC» для завершение игры (листинг 2.1).

Листинг 2.1. GDScript «Main.gd».

```
1   extends Node3D
2
3   @export var fast_close := true
4
5   func _ready() -> void:
6           Input.set_mouse_mode(Input.MOUSE_MODE_CAPTURED)
7           if !OS.is_debug_build():
8                   fast_close = false
9           if fast_close:
10                  print("** Fast Close enabled in the 'L_Main.gd' script **")
11                  print("** 'Esc' to close 'Shift + F1' to release mouse **")
12          set_process_input(fast_close)
13
14  func _input(event: InputEvent) -> void:
15          if event.is_action_pressed(&"ui_cancel"):
16                  get_tree().quit() # Quits the game
```

Строка 3. Если правда, то включается обработку ввода (строка 12), а 9 строке уведомляет через окно отладки пользователя, клавиша ESC для завершения игры.

Строка 6. Перехватывает курсор в центр позиций окно и скрывает его курсора.

Строка 7 — 8. Выключает обработку ввода если игра запущена без отладки.

Строка 14 — 16. Ожидает обработка ввода, клавиша ESC для завершения игры.

Мы создали объект CSGBox3D и назвали «Plane», задали Size x: 67, y: 1, z: 50. И включили столкновение «Use collision» на On.

Рисунок 2. Текстура

В панели свойства, мы создали материал (GeometryInstance3D → Geometry → Material Override), и задали значение цвет (Albedo → Color) #abababab, и изображение текстуру перетащили в Albedo → Texture и задали значение UV1 → Scale x: 50, y: 50, z: 50 (рис 3).

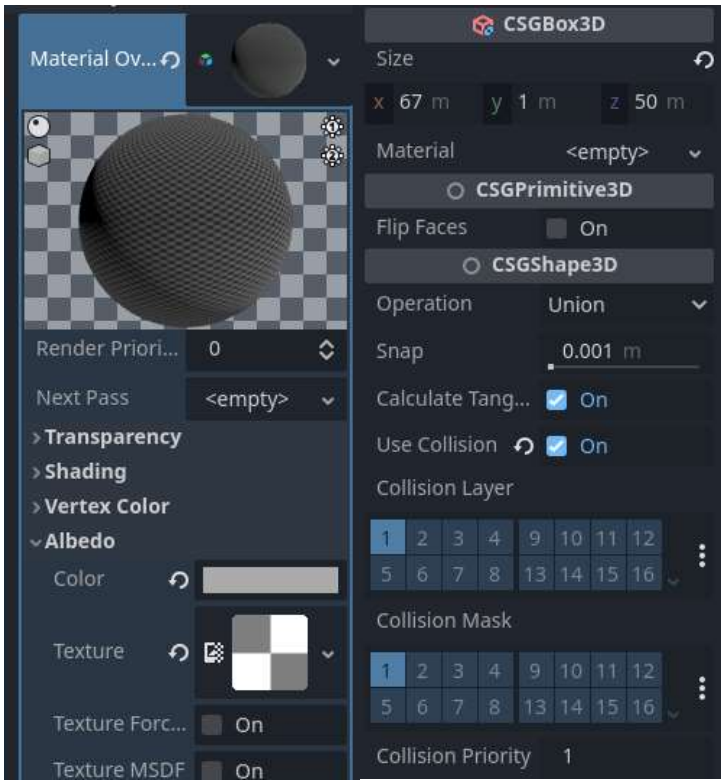

Рисунок 3. Заданные значение объекта «Plane».

Мы создали атмосферу в игре, создав объекты «Env» по типу Node, и объект WorldEnvironment и DirectionalLight3D

В WorldEnvironment, мы задали Background → Mode в значение Sky, и в sky → Sky, в Sky Material создали «Procedural Sky Material», а в «Procedural Sky Material» задали sky → Top Color в #62748c. Это задает внешний вид небо.

В WorldEnvironment, в свойстве Glow мы включили (Enable на on) это включает эффект свечение.

В DirectionalLight3D задали значение Transform → Rotation x: -60, y: -90, z: -90. Это направление источник света.

Мы создали объект CharacterBody3D назвали «Player» и сохранили в файл «res://Player/Player.tscn» и задали свойство Transform → Position x: -1, y: 2, z: 3.5 (рис 4).

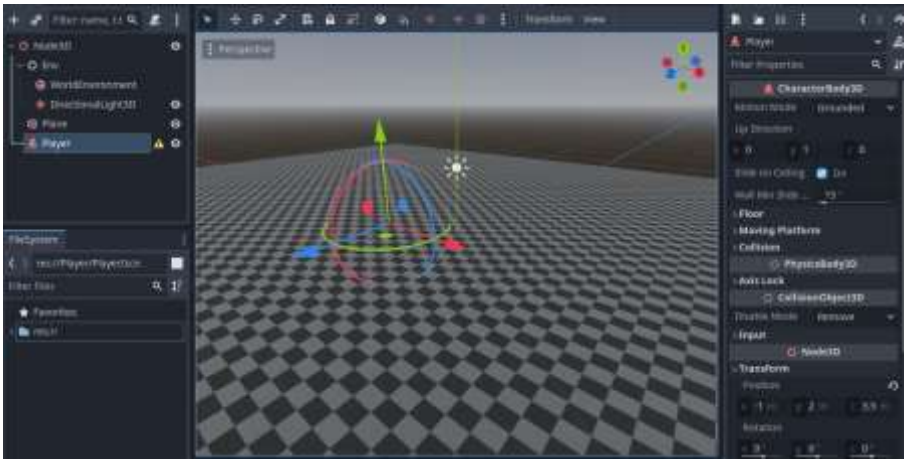

Рисунок 4. Главная сцена «Main.tscn».

Для столкновения, мы создали объект CollisionShape3D и назвали «Collision» и задали в свойстве Shape создали «CapsuleShape3D».

Создали объект Node3D и назвали «Head» и задали Transform → Position x: 0, y: 0.64, z: 0 в нем создали под объект Camera3D и назвали его «Camera». Создали Node и назвали «Sprint» (рис 5).

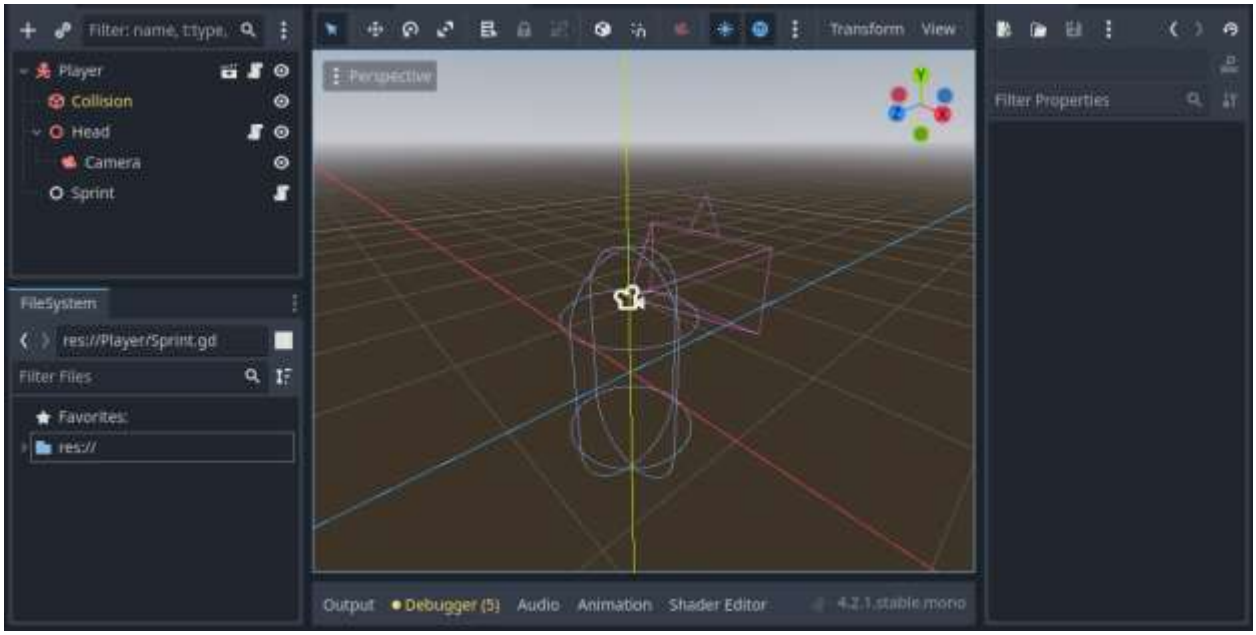

Рисунок 5. Объект игрок «Player.tscn».

Чтобы настроить управление клавиатуры игрока мы добавили в список move_back, move_forward, move_left, move_right — управление перемещение игрока, jump — прыжок игрока, sit — спуск вниз игрока, sprint — ускорение игрока (рис 6).

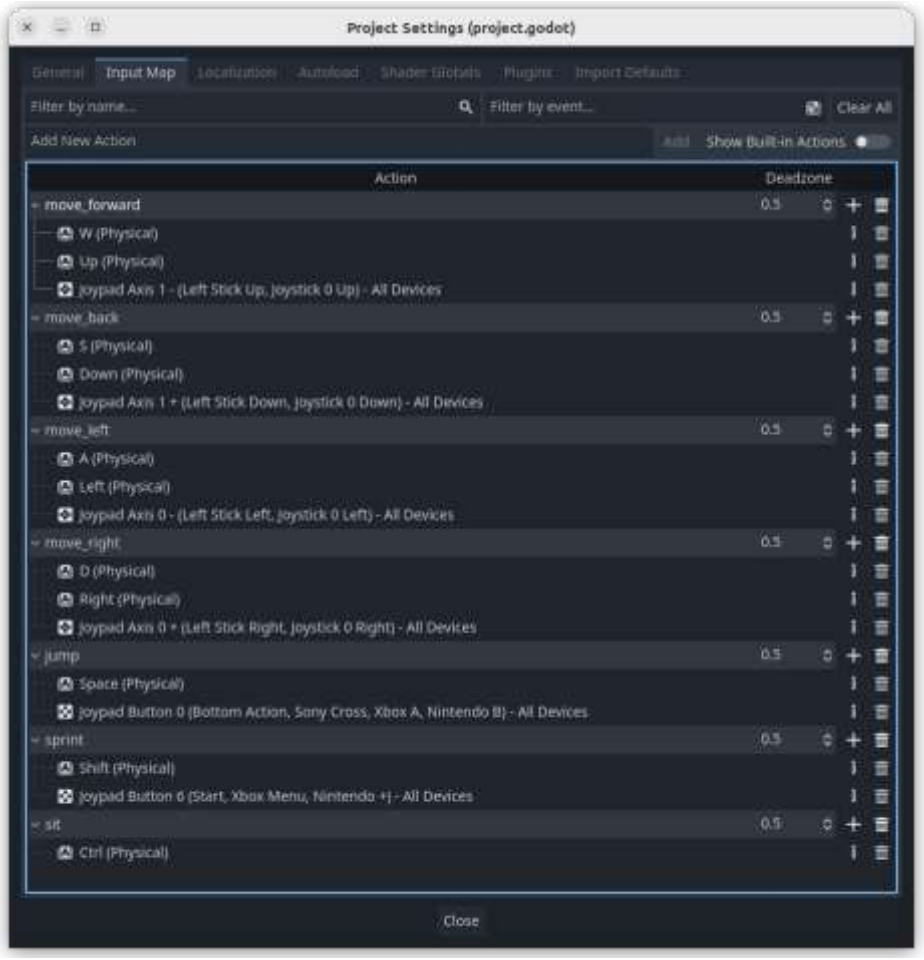

Рисунок 6. Окно настройки управление контроллера игрока

3 Выводы

В данное статьи была создана заготовка игры, где содержится объект игрока и контроллер управление и текстура пола В результате работе было представлено заготовка, пол, игрок, объект.

Библиографический список

1. Salmela T. Game development using the open-source Godot Game Engine. 2022.
2. Andersson C., Mellander T. A 2D Game Rewind-System Using Godot Game Engine, Performance Comparison and Analysis. 2021.
3. Mäkelä H. Development of a 3D mahjong video game in Godot Engine. 2021.