

Таначева Е. Г.
студент
НТГСПИ (филиал) УрГПУ
Россия, Нижний Тагил

Научный руководитель:
Бужинская Н.В.
к.п.н., доцент кафедры ИТФМ
НТГСПИ (филиал) УрГПУ
Россия, Нижний Тагил

ПРОГРАММИРОВАНИЕ В СРЕДЕ ROBLOX STUDIO

Аннотация

Roblox пользуется популярностью во всем мире. Однако многие пользователи воспринимают ее как компьютерную игру, но на самом деле в среду встроены инструменты для самостоятельного программирования игр на языке Lua. В данной статье представлены ключевые этапы создания игрового мира в среде Roblox Studio, а именно процесс разработки дизайна и программирование функционала.

Ключевые слова: игровой мир, среда, программирование, Roblox Studio, Lua.

Tanacheva E. G.
student
NTGSPI (branch) of USPU
Russia, Nizhny Tagil
Scientific supervisor:
Buzhinskaya N.V.
Candidate of Pedagogical Sciences,
Associate Professor of the Department of ITFI
NTGSPI (branch) of USPU
Russia, Nizhny Tagil

PROGRAMMING IN THE ROBLOX STUDIO ENVIRONMENT

Annotation

Roblox is popular all over the world. However, many users perceive it as a computer game, but in fact, the environment has built-in tools for self-programming games in the Lua language. This article presents the key stages of creating a game world in the Roblox Studio environment, namely the design development process and functionality programming.

Keywords: game world, environment, programming, Roblox Studio, Lua.

Компьютерные игры предоставляют возможность учиться программированию в интерактивной и увлекательной форме. Учащиеся

сталкиваются с различными задачами и проблемами, которые требуют творческого подхода и умения анализировать информацию.

Одной из компьютерных игр, способствующих развитию навыков алгоритмического мышления, пространственного воображения критического мышления является Roblox, которая является крупнейшей развлекательной платформой, на которую заходит несколько миллионов активных пользователей в месяц [1]. Данная игровая платформа, предлагающая уникальный опыт для пользователей, объединяет в себе увлекательный игровой мир с визуальным конструктором, позволяя учащимся разрабатывать персонажей, создавать игровые миры, используя 3D модели и скрипты, написанные на языке программирования Lua.

Скрипты в Roblox Studio пишут на встроенном языке программирования Lua. Lua – язык программирования расширений, разработан для поддержки общего процедурного программирования с возможностью описания данных. Lua реализован как библиотека, написан на чистом C, общее подмножество стандартного C и C++ [6].

Этот язык очень уникален, потому что это один из немногих текстовых языков, предназначенных для образовательных целей. Этот язык позволяет вам набирать ваши программы так, чтобы вы чувствовали, что на самом деле пишете код, но часть тяжелой работы выполняется за вас. С Lua вам придется беспокоиться об орфографии и пунктуации, но вам не придется беспокоиться о таких вещах, как создание механики прыжков или реалистичной физики.

Для того, чтобы показать возможности визуального программирования нами разработан игровой мир в среде Roblox Studio.

В основе игры будет находиться лабиринт, который позволит обучать детей запоминать пройденный маршрут, собирая монеты.

Требования к игре:

- разработать схему лабиринта;
- изменить интерфейс;
- создать скрипт по сбору монет;
- создать магазин с товарами;
- прописать скрипт, который будет выдавать товар по определенной

цене.

Первый этап разработки включает выбор локации, построение лабиринта и выявления мест, в которых будут располагаться монеты.

Чтобы начать программировать в Roblox, необходимо установить среду программирования. Для этого нужно зайти на сайт www.roblox.com, зарегистрироваться и дальше мы попадем на страницу нашего профиля.

Нажимаем на вкладку «Create», далее внизу располагается кнопка «Start Creating», после нажатия которой необходимо скачать и установить приложение.

Затем, открыв данное приложение, нужно ввести логин и пароль, которые были указаны при регистрации. После всех выполненных действий, мы сможем войти в систему и приступить к созданию мира в Roblox Studio.

После входа в Roblox Studio перед нами появляются шаблоны для создания игр. Для лучшего понимания интерфейса рассмотрим его на примере карты «Flat Terrain» (рис. 1).

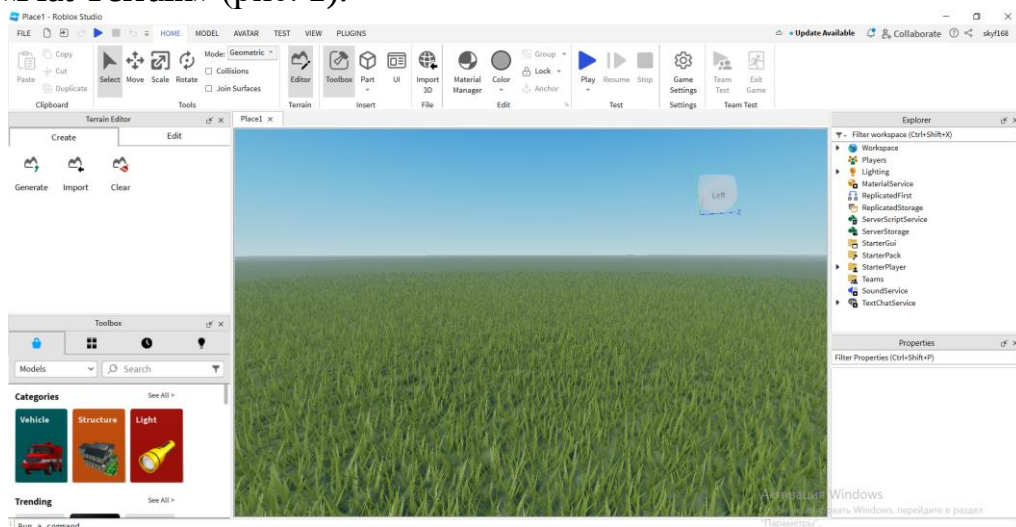


Рис.1. Интерфейс

В центре экрана располагается игровая сцена. Справа находится компонент «Explorer» (рис. 2), в котором располагаются все сервисы, необходимые для настройки уровня. Самый важный из них «Workspace» – сервис, внутри которого находятся названия всех объектов, добавленных на игровую сцену. Чтобы раскрыть этот список, необходимо нажать на стрелку, слева от него.

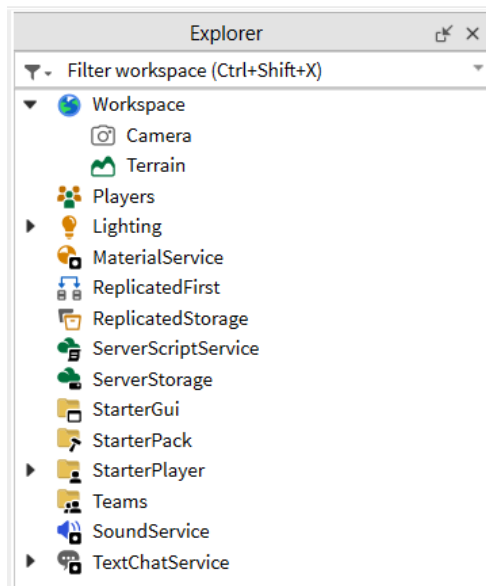


Рис. 2. Меню Explorer

Если нажать на название любого объекта, можно увидеть, как в правом нижнем углу появилось окно Properties, т.е. «Свойства». В данном окне можно посмотреть и изменить свойства выбранного объекта (рис. 3).

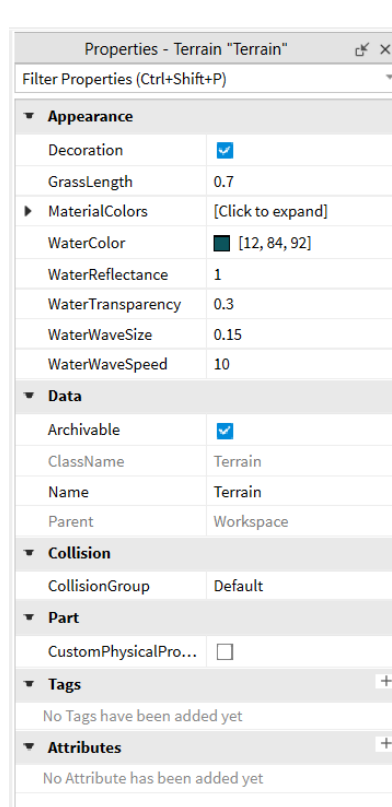


Рис. 3. Properties

В левом верхнем углу располагается «Toolbox», в котором находятся различные модели и объекты, созданные игроками или разработчиками. Эти объекты можно использовать для создания собственных игр. Необходимые объекты можно искать по названию в поисковой строке, но т.к. у сервиса нет перевода, нужно писать его на английском. Например, если нам необходимо найти дерево, нужно ввести «Tree» и выбрать необходимую модель (рис. 4).

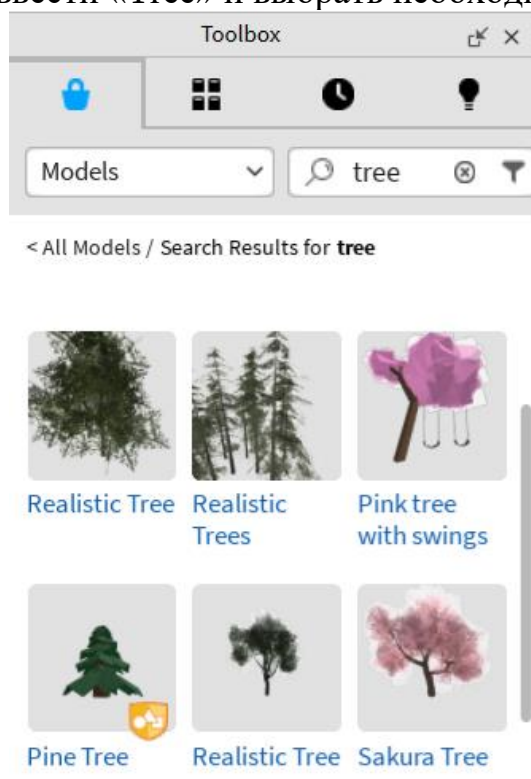


Рис. 4. Toolbox. Поиск объекта

Вверху размещена «панель инструментов» или «Toolbar» (рис. 5). Здесь наиболее важными для нас будут режимы работы с объектами, и кнопки, запускающие игру.

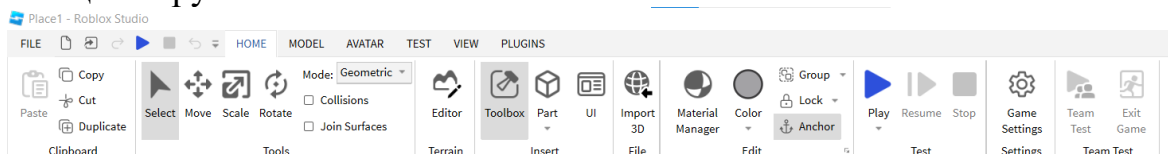


Рис. 5. Toolbar

Построение лабиринта началось с создания «Lobby» (место, где появляется игрок, когда заходит в игру), после была создана площадка, на которой расположился сам лабиринт. Создание происходило при помощи нажатия на панели инструментов кнопки «Part» и выбран инструмент «Block». Стенки лабиринта возводились при помощи тех же самых действий (рис. 6).

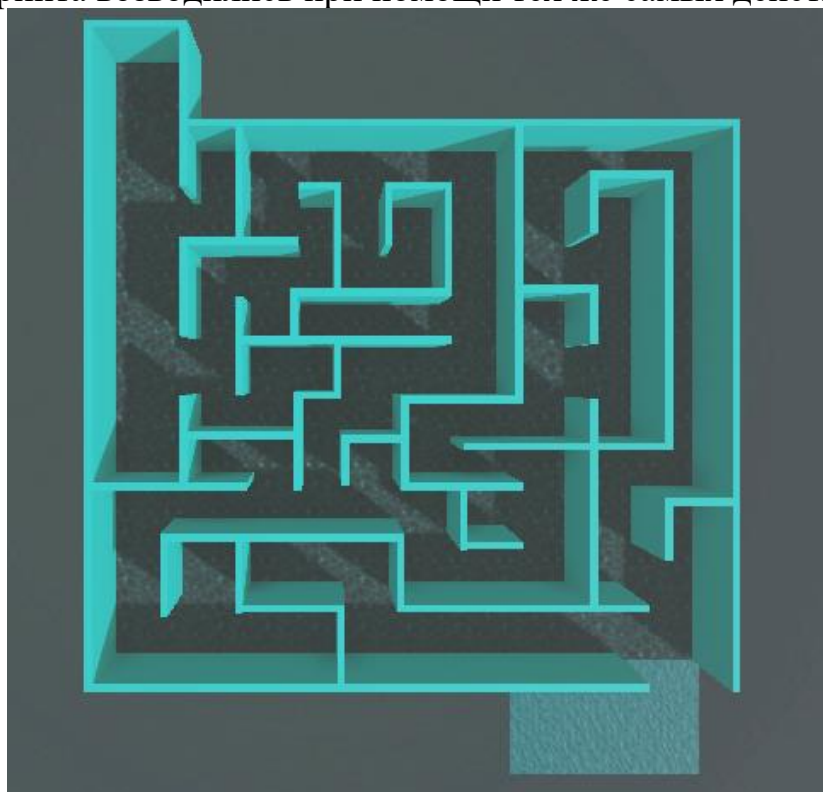


Рис. 6. Схема лабиринта

После создания необходимо было поработать с интерфейсом. Для этого сначала был изменен пол, на котором расположен лабиринт. Это можно сделать при помощи инструментов «Material Manager» и «Color» в панели инструментов. Первый инструмент поменяет материал, а второй расцветку. Пол сделан из «Cracked Lava» в пастельно-оранжевом оттенке, а стены лабиринта выполнены в таком же оттенке, но материал был выбран другой – «Pebble».

Чтобы игра не казалась пустой, нужно выбрать фон. Для этого в панели «Toolbox» в поисковой строке была вбита фраза «sky» (рус. «небо») и из большого списка была выбрана необходимая модель, которая подходила бы к стилистике построенного лабиринта и создавала необходимую атмосферу.

Однако в это же время она не должна отвлекать игрока от самого процесса прохождения, поэтому была выбрана модель «Aesthetic sky» (рис. 7).

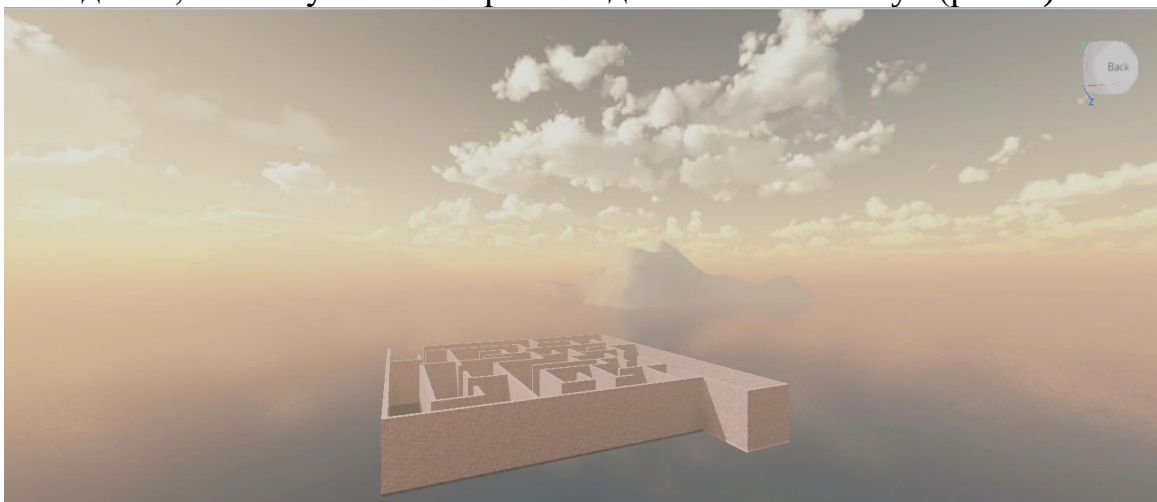


Рис. 7. Созданный лабиринт с измененным интерфейсом
После в игре были созданы монеты и прописан следующий скрипт (рис. 8).

```
1  local function ld(player)
2      local leaderstats = Instance.new("Folder", player)
3      leaderstats.Name = "leaderstats"
4      local coin = Instance.new("IntValue", leaderstats)
5      coin.Name = "Money"
6
7  end
8
9  game.Players.PlayerAdded:Connect(ld)
```

Рис. 8. Скрипт по выведению количества монет

В первой строке прописывается создание переменной «ld» и внутри ставим обязательно переменную, которая в будущем поможет сделать чтобы функция относилась именно к локальному игроку, который в данный момент зашел на сервер.

Во второй строке создаем дополнительную переменную «leaderstats» и прописываем команду `Instance.new`, которая отвечает за создание папки и расположена она будет в самом игроке. Переименовываем эту папку в «leaderstats». Очень важно, чтобы она носила такое название иначе программа не поймет, что это за название.

В четвертой строке создается переменная с монетами через команду `Instance.new`. Нас интересует количество (или числовое значение), которое прописывается через `IntValue`. Следовательно, в папке `Folder` будет создан документ, который хранит значение цифры.

В пятой строке указываем название нашей монете – «Money».

Программа не будет работать, если мы не пропишем в какой момент она должна запускаться, поэтому в 9 строке мы указали, что она должна начать работать, когда игрок добавлен у нас прикрепляется функция счета собранных монет.

Теперь нужно добавить сами монеты, которые игрок будет собирать. Это можно сделать с помощью панели «Tollbox».

После выполнения скрипта должно получиться следующее: в правом верхнем углу высветится область, в которой будет показано имя зашедшего игрока (skyf168) и количество собранных монет (1) (рис. 9).



Рис. 9. Сработанный скрипт с выводением количества монет

Теперь перейдем к созданию магазина, в котором у игроков будет возможность потратить заработанные деньги на приобретение товаров. Для этого нам необходимо использовать GUI, инструмент, с помощью которого можно разместить элементы на экране.

Для того, чтобы создать такие кнопки, необходимо в «Explorer» найти «StarterGui», нажать на «+» справа от него, выбрать «ScreenGui», а после нажать на Frame. Далее появится область на экране – это место, на котором будет находиться наш будущий магазин.

На следующем этапе добавляем кнопку, которая будет открывать магазин. Её добавляем в «ScreenGui» как «Frame» и размещаем под магазином. Теперь можно зайти с свойства и изменить размер текста, цвет, шрифт, заливку и т.д. Кроме этого, можно добавить рамку объекту, залить ее цветом

Перейдем к добавлению кнопок в магазин. Используем «TextBotton» и добавляем необходимое количество кнопок, которые хотим запрограммировать на получение «приза» за прохождение лабиринта. После необходимо все созданные кнопки перенести в Frame, иначе они не будут работать (рис. 10).

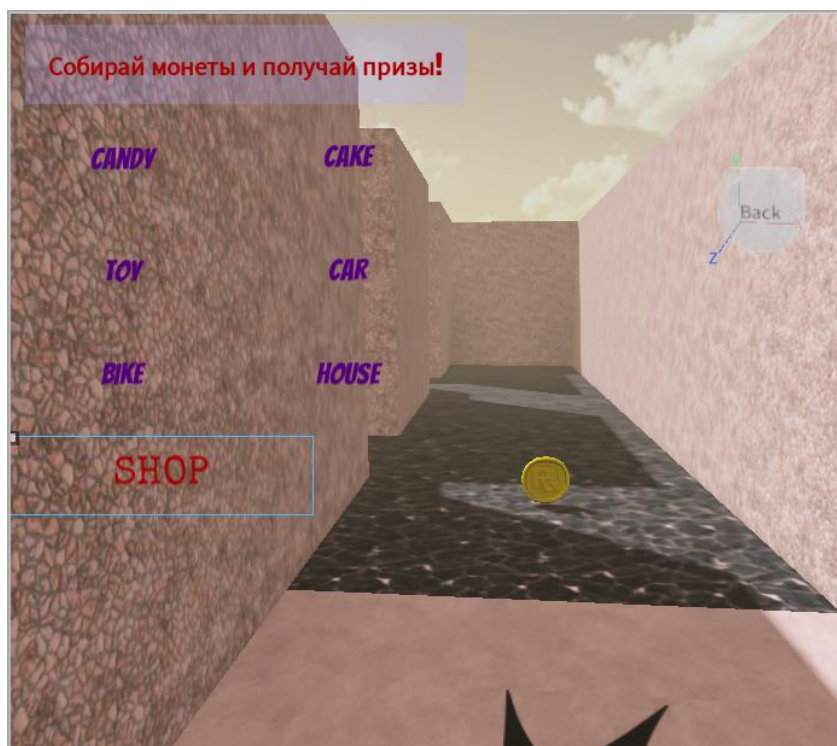


Рис. 10. Панель магазина с товарами

Теперь добавляем товары. Для этого в панели «ToolBox» ищем по названиям необходимые вещи и добавляем их в «ServerStorage». После в папке «ReplicatedStorage» необходимо создать «приглашения» (event) к созданным вещам. Нужно нажать на «+» и добавить шесть объектов RemonteEvent, одноименных с теми продуктами, которые были добавлены ранее.

Перейдем к добавлению пути, проводящему к кнопкам магазина. Для этого необходимо зайти в «StarterGui – Frame – Cake» и добавить к нему «LocalScript», в котором пропишем путь кнопки, т.е. при нажатии на кнопку она должна переносить объект на локацию. Эти действия необходимо прописать для каждой позиции в магазине. Напишем следующий скрипт для товара «Cake» (листинг 1).

Листинг 1

Скрипт для пути кнопки Cake

```
script.Parent.MouseButton1Click:Connect(function()
    game.ReplicatedStorage.Tools.Cake:FireServer()
end)
```

В «ServerScriptService» добавим скрипт, который будет выдавать товары за монеты. Данный скрипт прописан так, что, когда игрок набирает определенное количество монет, тогда он может приобрести на это сумму конкретный товар.

Таким образом, создание собственной игры — это увлекательный процесс, который позволяет проявить креативность и развить навыки программирования. Это занятие подходит как для детей, так и для взрослых, и может стать отличным хобби или даже профессией в будущем.

ЛИТЕРАТУРА

1. Брамбо, Зандер. Программирование в Roblox. Сделать игру проще простого: создание игр с помощью Roblox Studio и языка программирования Lua от «А» до «Я». М: ДМК-Пресс, 2022. 197 с.
2. Васильев А. А., Печатнова Ю. В. Термин «компьютерная игра»: опыт междисциплинарного анализа // Пролог: журнал о праве. 2021. №-2 (30). С 131-138/ URL: <https://cyberleninka.ru/article/n/termin-kompyuternaya-igra-opyt-mezhdistsiplinarnogo-analiza> (дата обращения: 20.05.2024).
3. Ерогова Инна Сергеевна Компьютерные игры в обучении // Вестник науки и творчества. 2016. №-9 (9). С. 17-21. URL: <https://cyberleninka.ru/article/n/kompyuternye-igry-v-obuchenii> (дата обращения: 23.05.2024).
4. Журнал Roblox. URL: <https://www.rbc.ru/magazine/2018/11/5bc4a1e59a794797cbc44fee> (дата обращения: 20.05.2024).
5. Центр создания Roblox. URL: <https://create.roblox.com> (дата обращения: 20.05.2024).
6. Lua 5.3 Руководство. URL: https://lua.org.ru/contents_ru.html (дата обращения: 20.05.2024).