

Разработка игрового приложения «Math» в MIT App Inventor: блочное программирование и реализация игровой механики

Екимова Яна Сергеевна

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье представлен процесс реализации программной логики математической игры «Math» в среде визуального программирования MIT App Inventor. В работе описана методология блочного программирования игровой механики, включая создание и инициализацию глобальных переменных для хранения слагаемых и правильного ответа, использование блока генерации случайных чисел для формирования примеров в диапазоне от 1 до 100, разработку процедуры setQuestion для автоматического обновления содержимого экрана, а также реализацию условного оператора для проверки введённого пользователем ответа с выводом всплывающих уведомлений через компонент Notifier. В результате создано функциональное мобильное приложение, генерирующее случайные арифметические примеры, предоставляющее интерактивную обратную связь при правильном или неправильном ответе и обеспечивающее переход к следующему вопросу, что подтверждает эффективность предложенного подхода к разработке обучающих игр в среде MIT App Inventor.

Ключевые слова: блочное программирование, mit app inventor, разработка игры, verticalarrangement, horizontalarrangement, математическая игра.

Development of the Math game application in the MIT App Inventor: block programming and implementation of game mechanics

Ekimova Yana Sergeevna

Sholom Aleichem Priamurskiy State University

Student

Abstract

The article presents the process of implementing the program logic of the mathematical game "Math" in the MIT App Inventor visual programming environment. The paper describes the methodology of block programming of game mechanics, including the creation and initialization of global variables to store terms and the correct answer, the use of a random number generation unit to generate examples in the range from 1 to 100, the development of the setQuestion procedure to automatically update the contents of the screen, as well as the implementation of a conditional operator to check the user-entered answer with the output of pop-up

notifications through the Notifier component. As a result, a functional mobile application was created that generates random arithmetic examples, provides interactive feedback on the correct or incorrect answer and provides a transition to the next question, which confirms the effectiveness of the proposed approach to developing educational games in the MIT App Inventor environment.

Keywords: block programming, mit app inventor, game development, vertical arrangement, horizontal arrangement, mathematical game.

1 Введение

1.1 Актуальность

Современное цифровое образование требует не только освоения визуального проектирования интерфейсов, но и глубокого понимания программной логики, лежащей в основе интерактивных приложений. Одним из ключевых этапов разработки является реализация алгоритмов обработки событий, генерации данных и проверки пользовательского ввода, что особенно важно при создании обучающих игр. Среда MIT App Inventor предоставляет возможность реализовать эту логику с помощью визуального блочного программирования, позволяя начинающим разработчикам освоить фундаментальные понятия программирования (переменные, условные операторы, процедуры, генерация случайных чисел) без написания текстового кода.

В условиях необходимости формирования у обучающихся алгоритмического мышления и навыков разработки полноценных игровых приложений, освоение этапа программирования игровой логики в MIT App Inventor приобретает особую значимость. Вторая часть разработки игры «Math» фокусируется на реализации вычислительной механики: от создания переменных и генерации случайных примеров до организации проверки ответов и системы обратной связи. Практическое описание процесса блочного программирования, включая создание процедур, обработку событий кнопок и условные переходы, позволяет системно представить завершающий этап создания функционального мобильного приложения, что делает данное исследование своевременным и востребованным для педагогов, методистов и разработчиков образовательного контента.

1.2 Обзор исследований

Б. Тасуов, Г. Муратова, Р. Сабирова описали MIT App Inventor как среду программирования, разработанную для создания приложений для мобильных устройств (смартфонов и планшетов) под управлением операционной системы Android. [1]. Рассмотрели возможности обучения школьников основам программирования, используя среду MIT App Inventor Н. Н. Самылкина, А.В.Етова [2]. А. Н. Вильданов Рассмотрел процесс разработки игры в жанре «Поиск предметов» для мобильных устройств на ОС Android с использованием MIT App Inventor [3]. Предложила использовать среду App Inventor для создания игр на мобильных устройствах в процессе обучения школьников программированию А. И. Малова [4].

1.3 Цель исследования

Целью исследования является реализация программной логики математической игры «Math» в среде визуального программирования MIT App Inventor, включающая создание и инициализацию глобальных переменных для хранения слагаемых и правильного ответа, разработку процедуры генерации случайных арифметических примеров, реализацию условного оператора для проверки введённого пользователем ответа, настройку компонента Notifier для вывода всплывающих уведомлений о правильности или ошибочности решения, а также обеспечение перехода к следующему вопросу для формирования полноценного игрового цикла и развития вычислительных навыков у пользователей.

2 Материалы и методы

Для реализации программной логики математической игры в качестве среды визуального блочного программирования был использован MIT App Inventor, который предоставляет встроенный редактор Blocks, позволяющий создавать алгоритмы обработки событий, управлять переменными, использовать условные конструкции и математические операции без написания текстового кода.

3 Результаты

Переходим на сайт MIT App Inventor <https://appinventor.mit.edu>, открываем ранее созданный проект «Math».

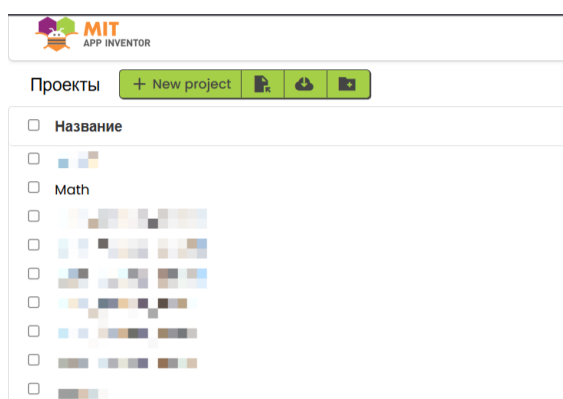


Рисунок 1. Открытие проекта

Переходим во вкладку «Blocks», чтобы перейти от визуального конструирования интерфейса к блочному программированию логики работы приложения.

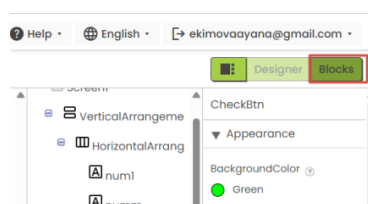


Рисунок 2. Переход на вкладку Blocks

Добавляем блоки из категории «Variables» для объявления переменных, необходимых для хранения значений слагаемых, и обращаемся к категории Math, чтобы использовать блок генерации случайных чисел, обеспечивающий случайный подбор примеров.

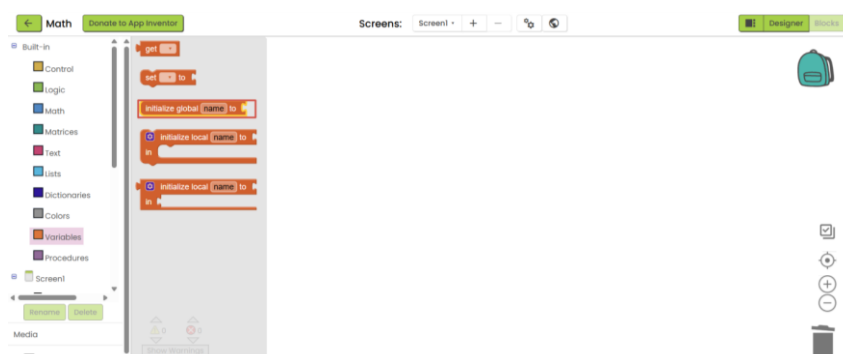


Рисунок 3. Добавление блока

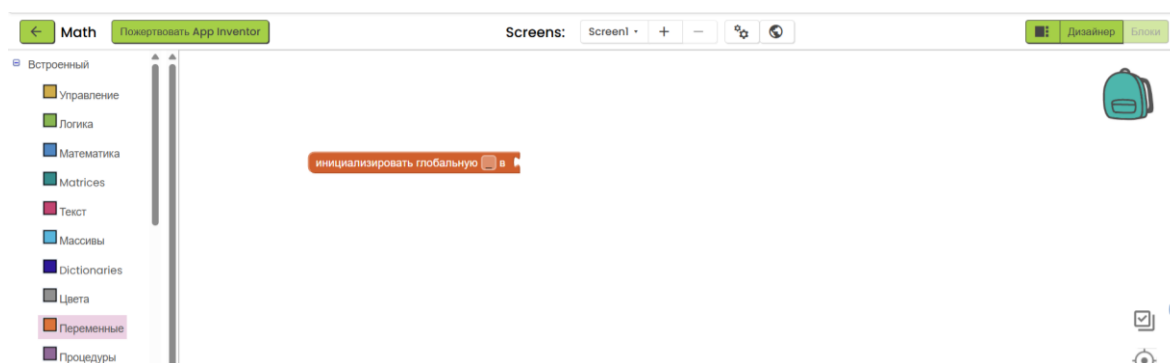


Рисунок 4. Добавление блока

Так же добавляем блок с числовым значением «0», который используется для инициализации глобальных переменных, задавая их начальное состояние перед запуском генерации случайных чисел для математического примера.

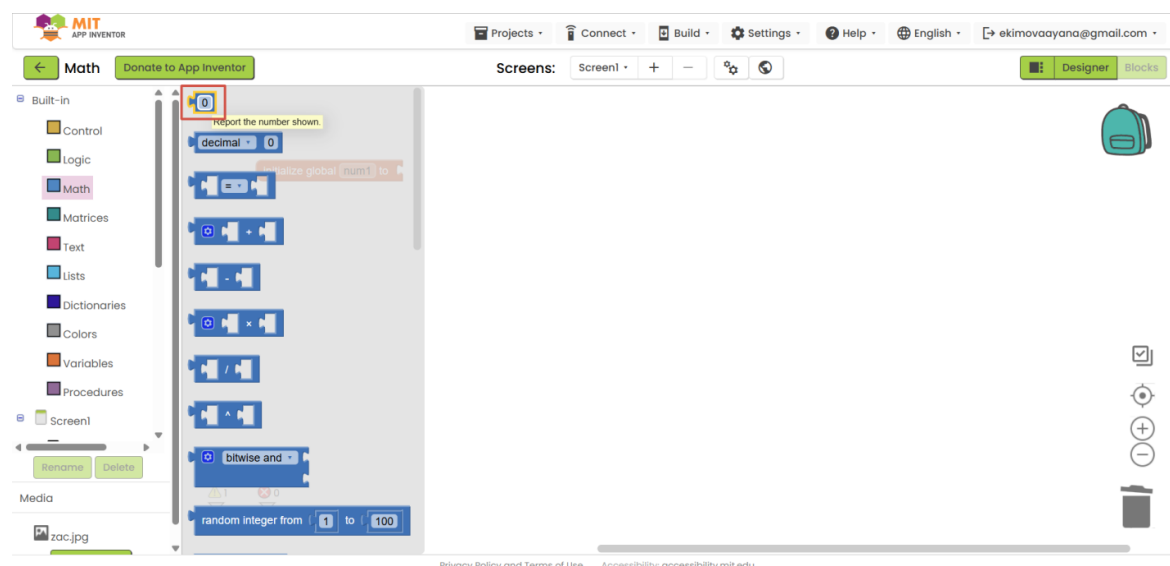


Рисунок 5. Добавление блока с числовыми значениями

Затем соединяем, переименовываем и копируем несколько раз блок инициализации глобальной переменной, создавая переменные «num1», «num2» и «answer» со значением 0, которые необходимы для хранения слагаемых и вычисления правильного ответа в программной логике приложения.

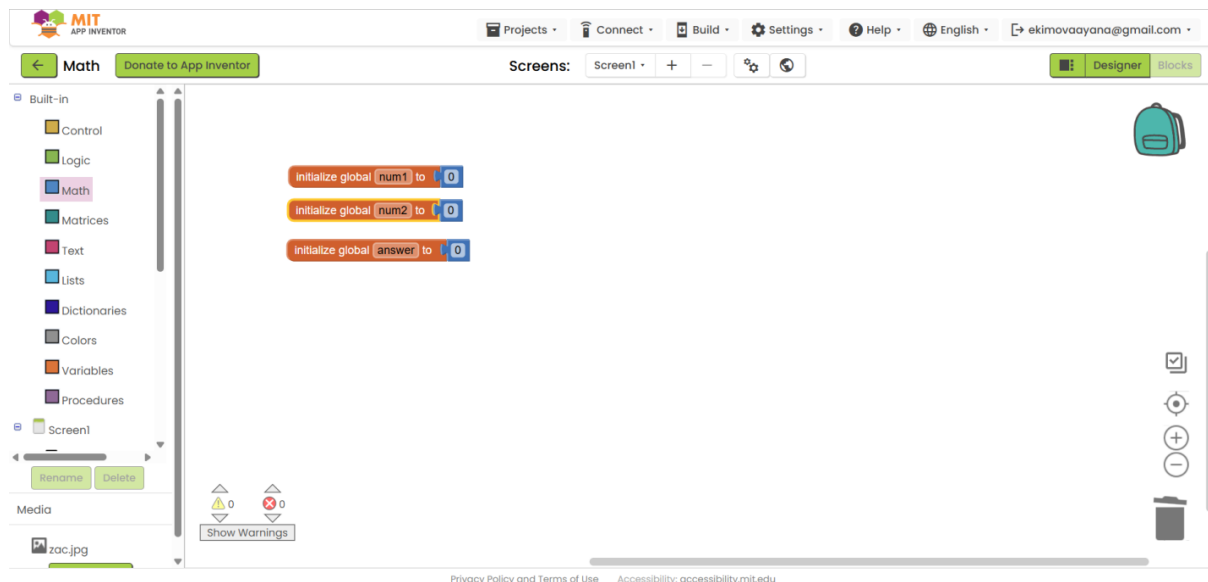


Рисунок 6. Создание переменных

Добавляем блок процедуры из категории «Procedures» и переименовываем его в «setQuestion», создавая отдельную функцию для генерации новых математических примеров.

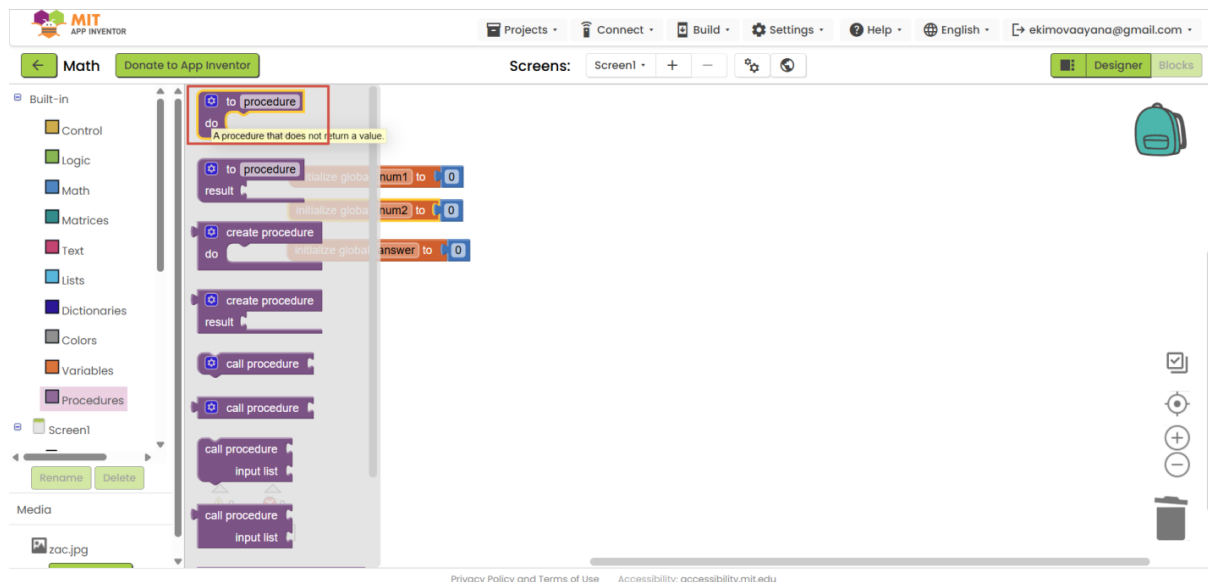


Рисунок 7. Добавление блока процедуры

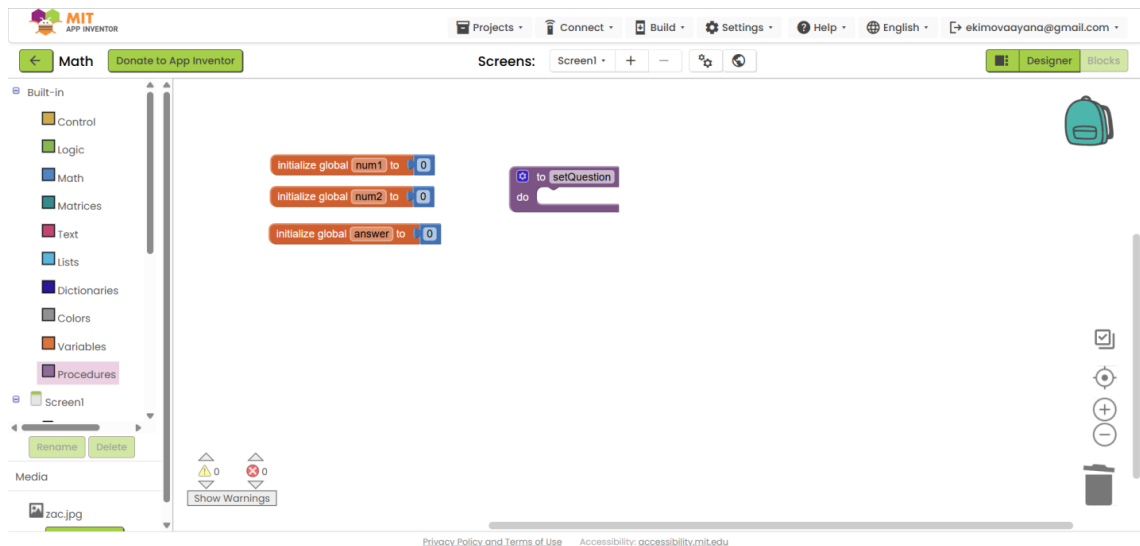


Рисунок 8. Добавление блока процедуры

Затем добавляем блок присваивания значения глобальной переменной num1 внутрь процедуры setQuestion для определения первого слагаемого в генерируемом математическом выражении.

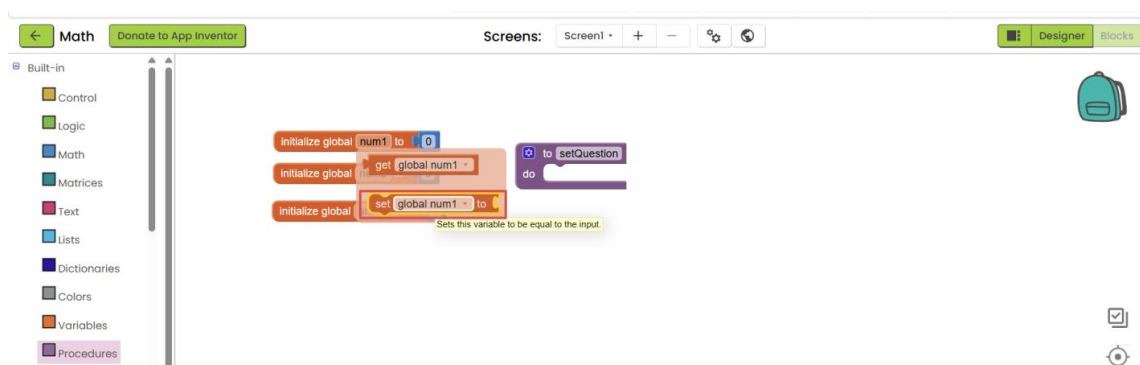


Рисунок 9. Добавление блока присваивания значения глобальной переменной

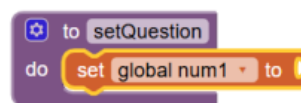


Рисунок 10. Добавление блока присваивания значения глобальной переменной

Добавляем блок «random integer from» из категории Math, задающий диапазон генерации случайных чисел от 1 до 100, и присоединяем его к блоку изменения значения глобальной переменной «num1», что позволяет случайным образом выбирать первое слагаемое для математического примера.

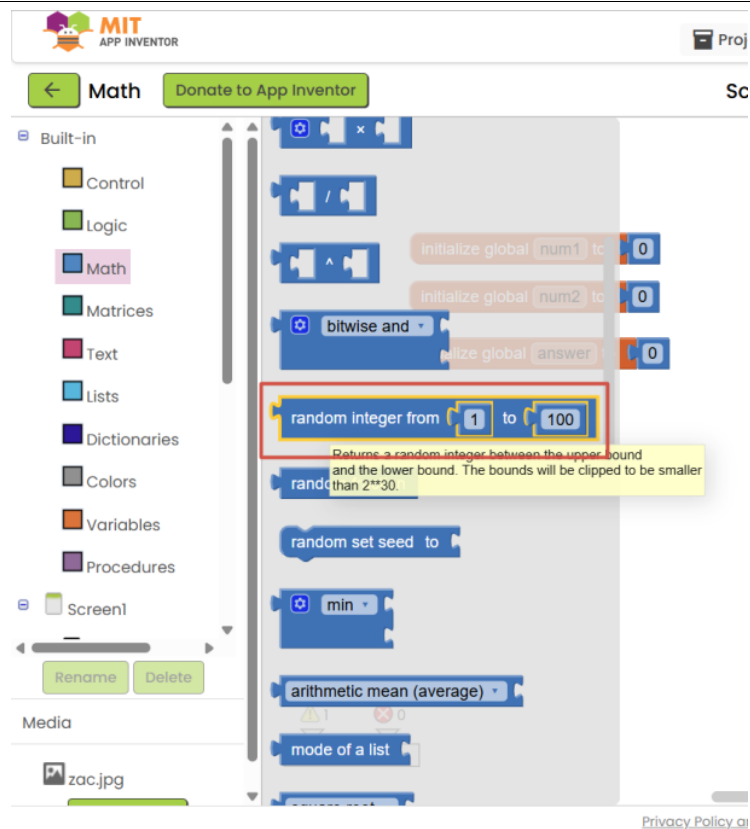


Рисунок 11. Добавление блока random integer from



Рисунок 12. Соединение блоков

Копируем и меняем переменную на «num2», добавляя в процедуру «setQuestion» второй блок присваивания, который генерирует случайное число от 1 до 100 для определения второго слагаемого в примере.

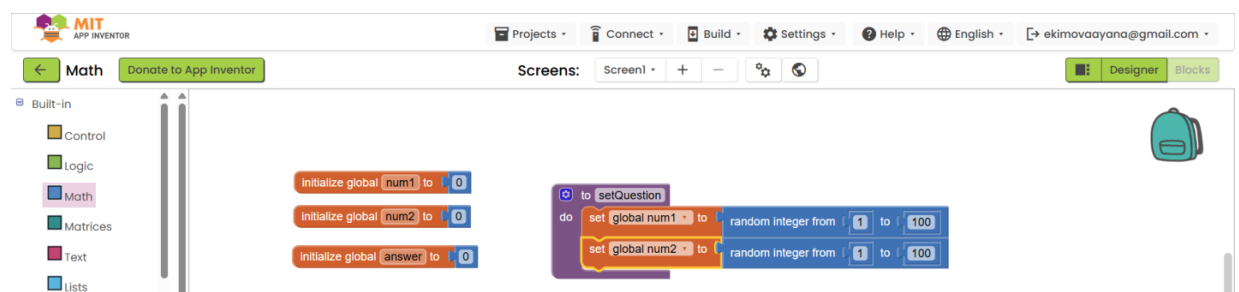


Рисунок 13. Добавление второго блока

Далее выбираем блок присваивания значения глобальной переменной «answer», необходимый для вычисления и сохранения правильного ответа,

который будет получен в результате сложения сгенерированных случайных чисел «num1» и «num2».

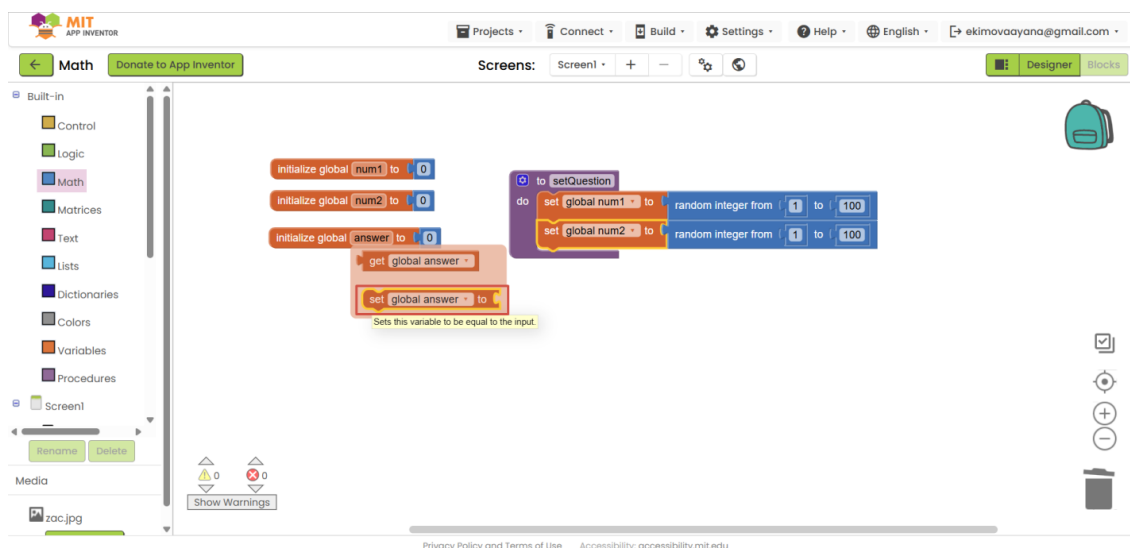


Рисунок 14. Присваивание переменной

Добавляем переменные «num1» и «num2» в блок математической операции сложения, объединяя их внутри процедуры «setQuestion» для вычисления правильной суммы, которая затем сохраняется в глобальной переменной «answer».

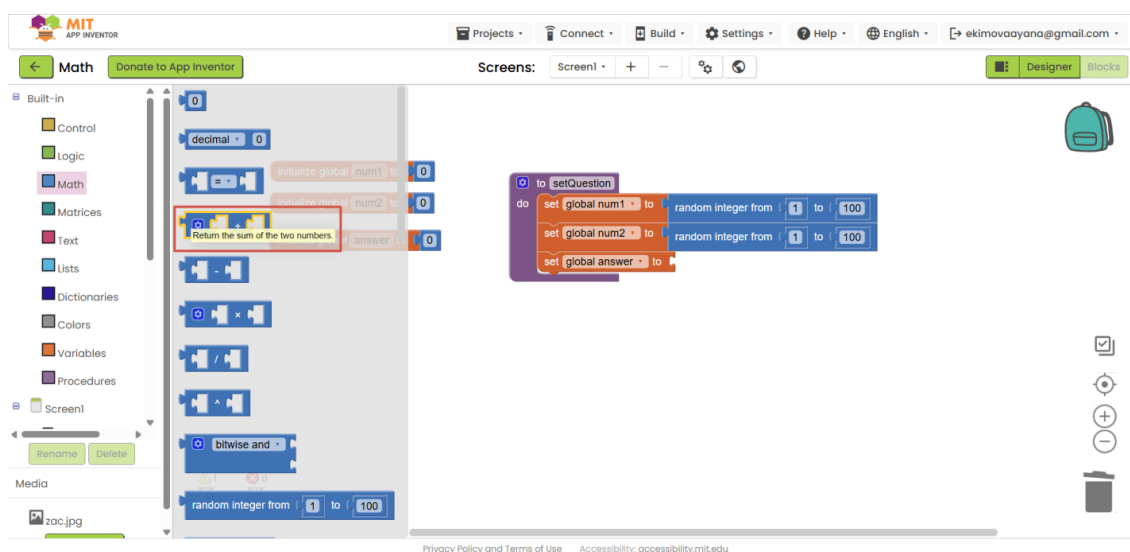


Рисунок 15. Добавление переменных

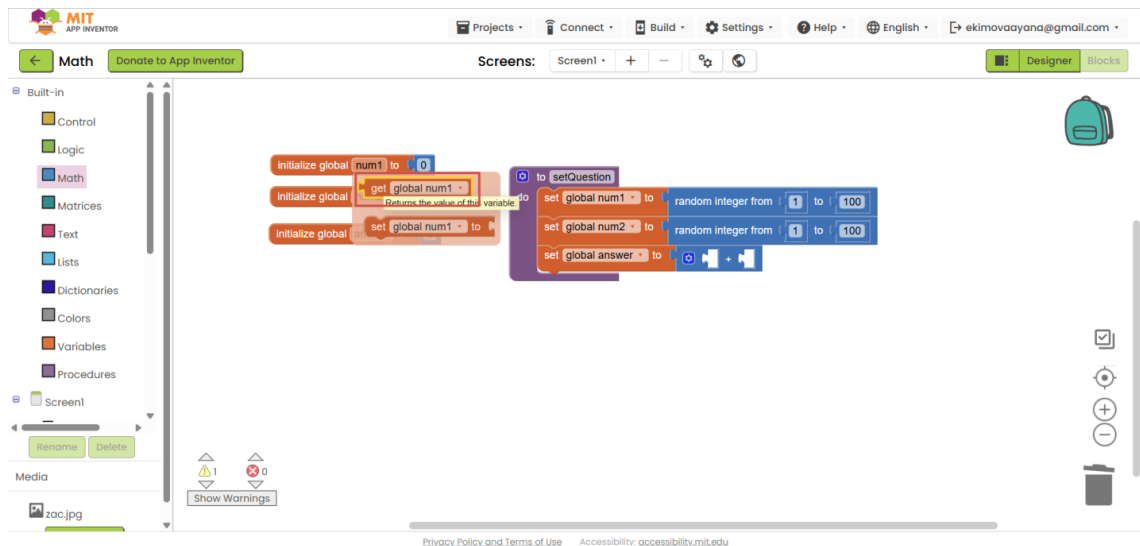


Рисунок 16. Добавление переменной num1

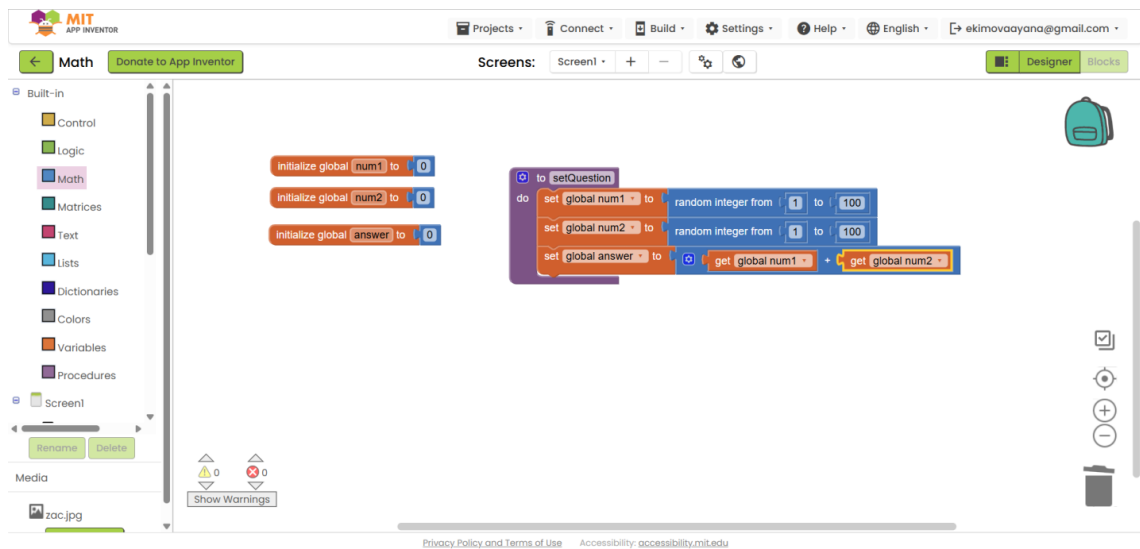


Рисунок 17. Добавление переменной num2

Затем переходим и выбираем компонент TxtBox, после чего перетаскиваем блок установки его текстового свойства (Text) в процедуру «setQuestion», чтобы очищать поле ввода при генерации нового примера.

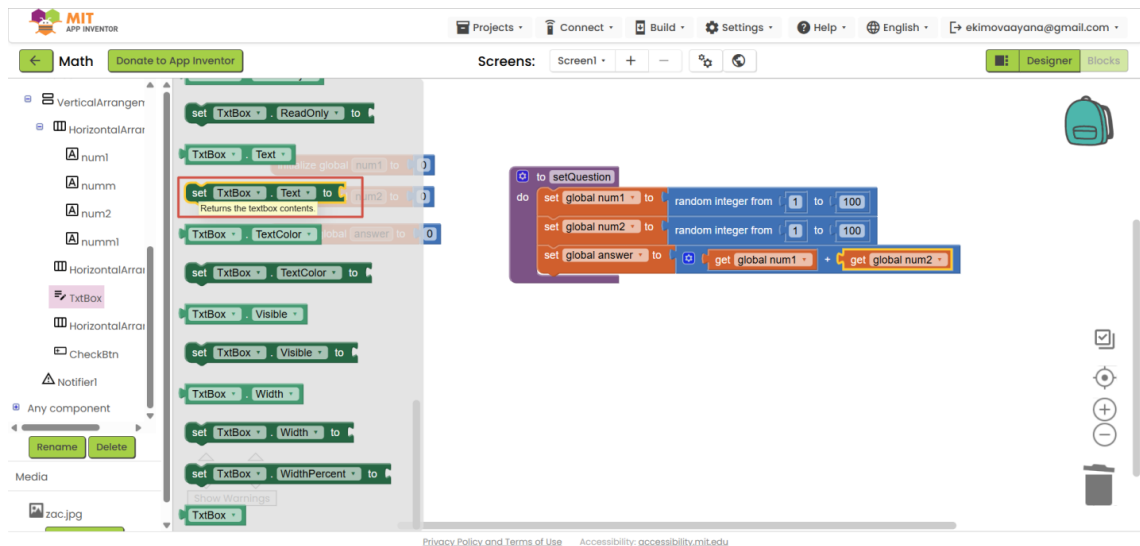


Рисунок 18. Настройка компонента TextBox

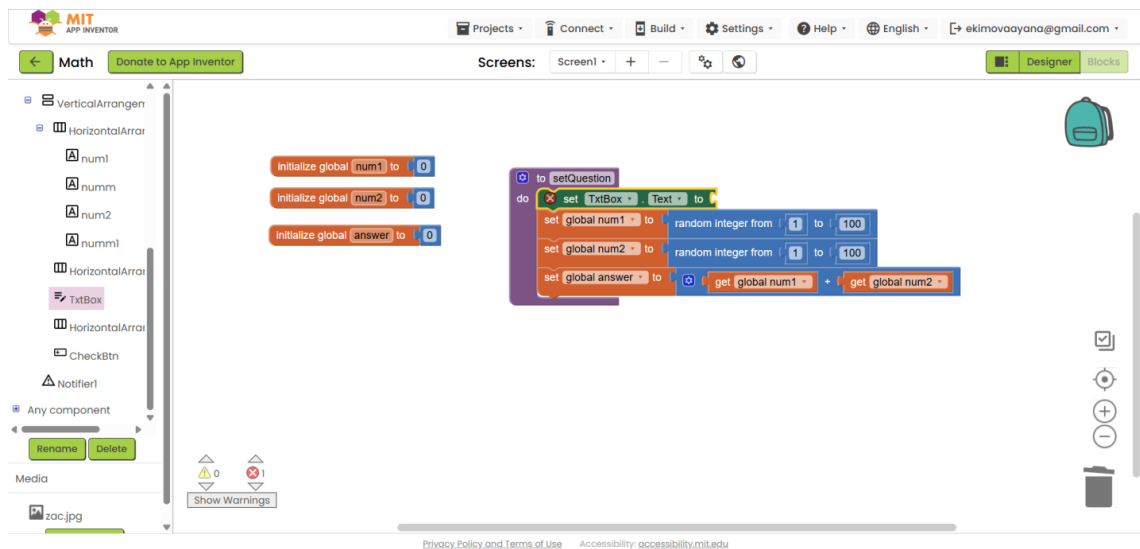


Рисунок 19. Настройка компонента TextBox

Добавляем блок пустой текстовой строки из категории Text в свойство Text компонента TextBox, чтобы очищать поле ввода перед генерацией нового примера.

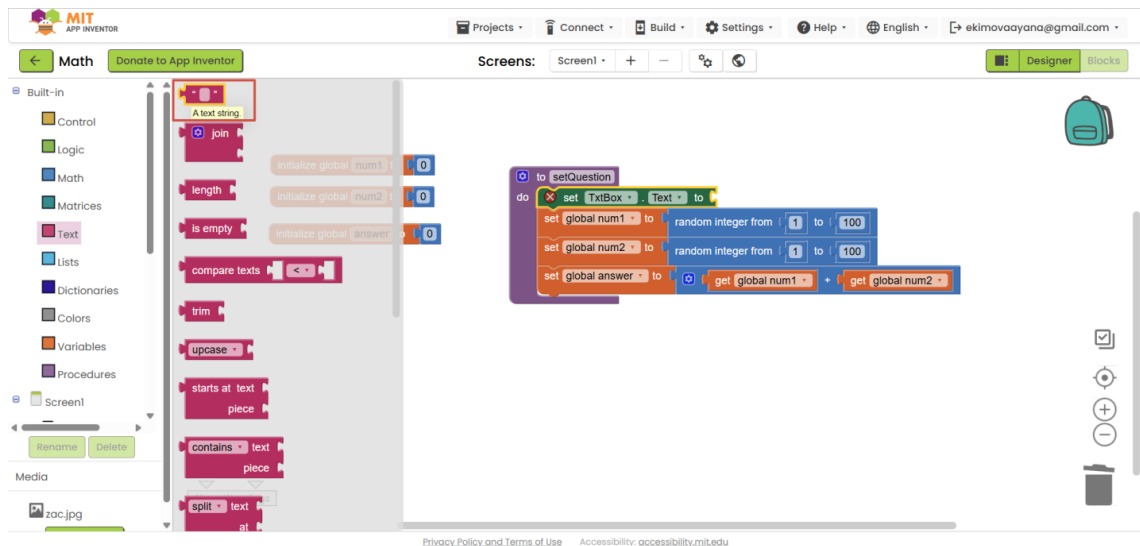


Рисунок 20. Выбор пустой текстовой строки

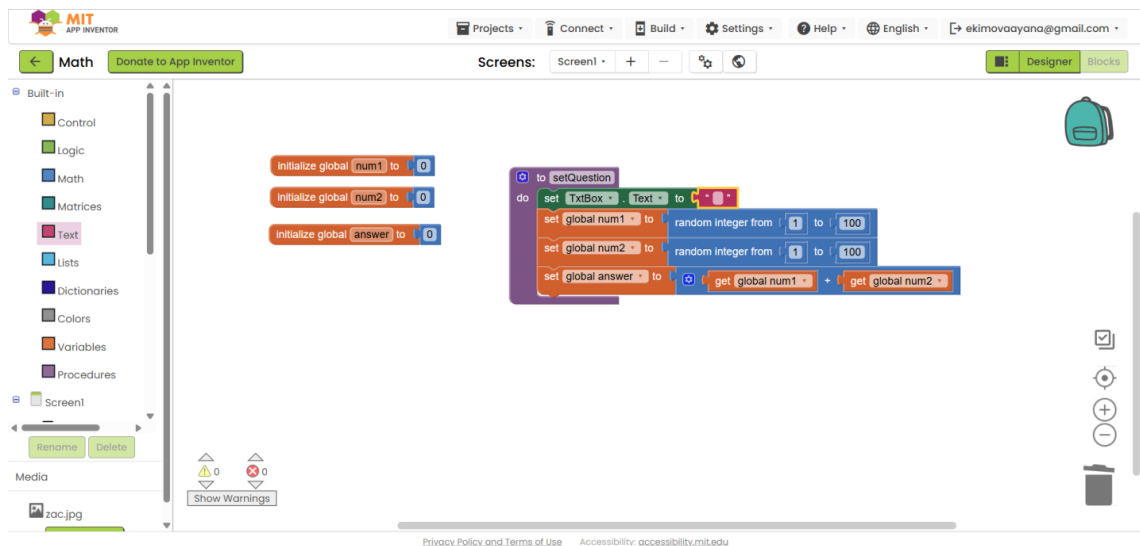


Рисунок 21. Добавление пустой текстовой строки

Затем добавляем блоки изменения свойства Text для компонентов «num1» и «num2», присваивая им значения соответствующих глобальных переменных, чтобы отобразить сгенерированные случайные числа в интерфейсе игры.

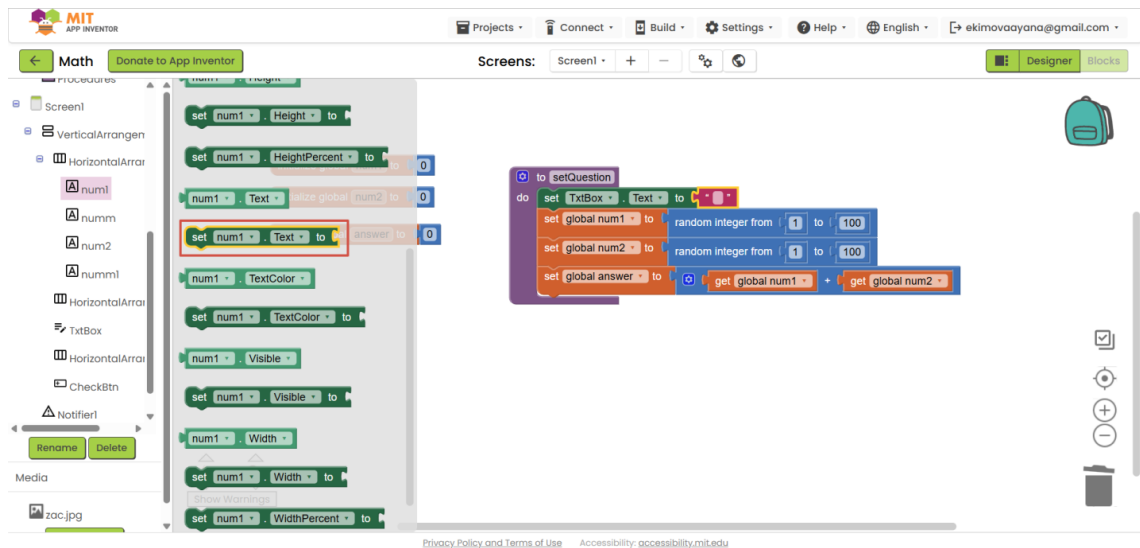


Рисунок 22. Добавление блоков изменения

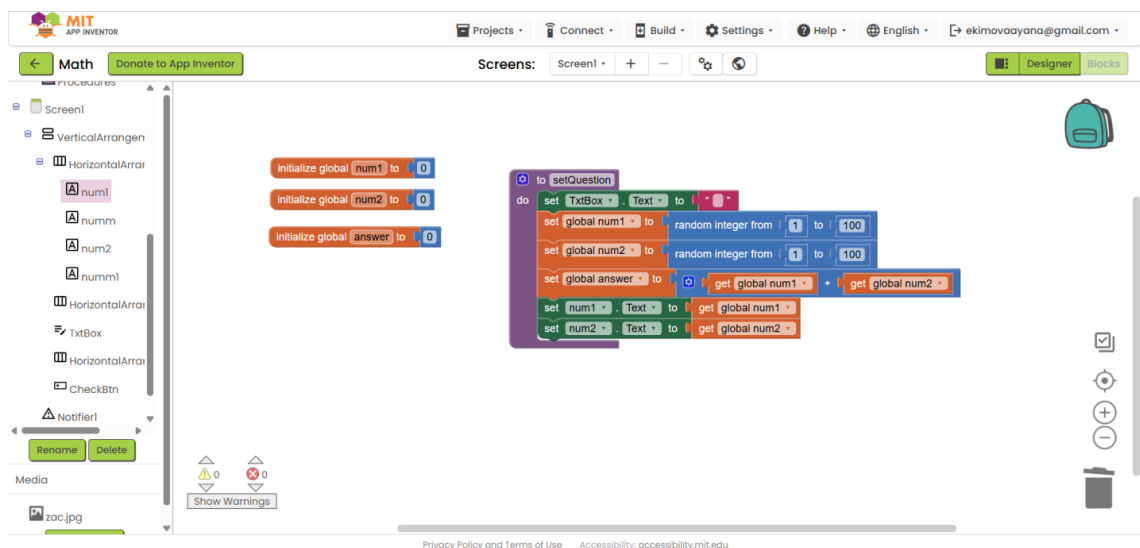


Рисунок 23. Добавление блоков изменения

Далее добавляем блок события «Screen1.Initialize», обеспечивающий выполнение кода при старте приложения, и размещаем внутри него вызов процедуры «setQuestion» для автоматической генерации первого математического примера сразу после запуска игры.

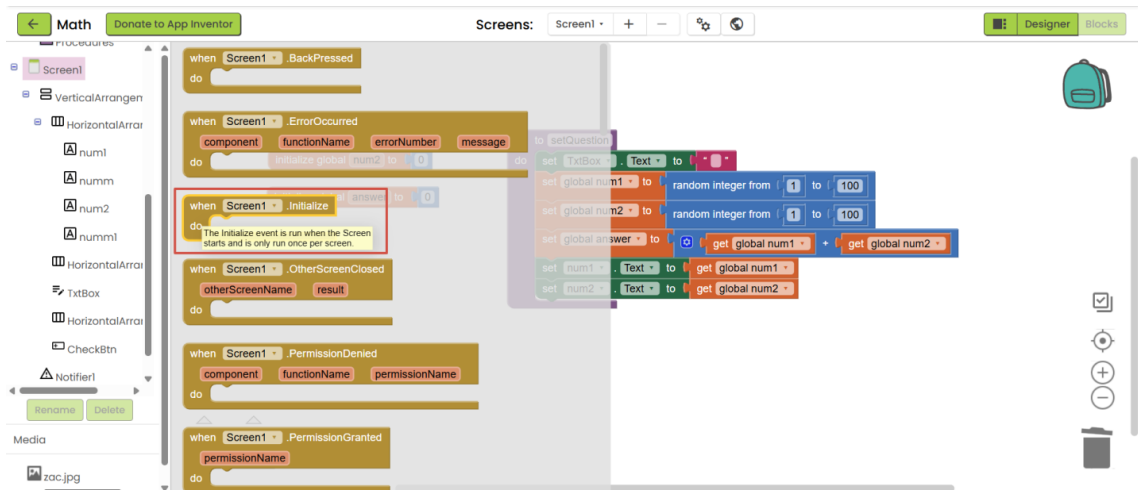


Рисунок 24. Добавление блока события

Затем берем блок вызова процедуры «call setQuestion» из категории Procedures и перетаскиваем его внутрь обработчика события «Screen1.Initialize», чтобы при запуске приложения сразу вызывалась функция генерации математического примера.

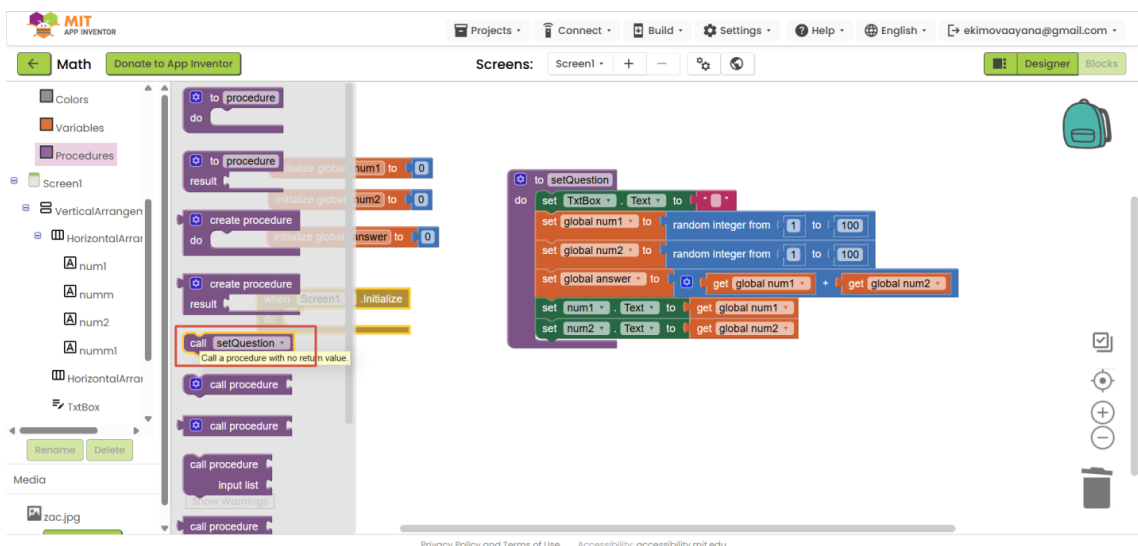


Рисунок 25. Добавление блока вызова

Добавляем блок события «when CheckBtn. Click» для кнопки проверки, который будет активировать логику сравнения введенного пользователем числа с правильным ответом при каждом нажатии.

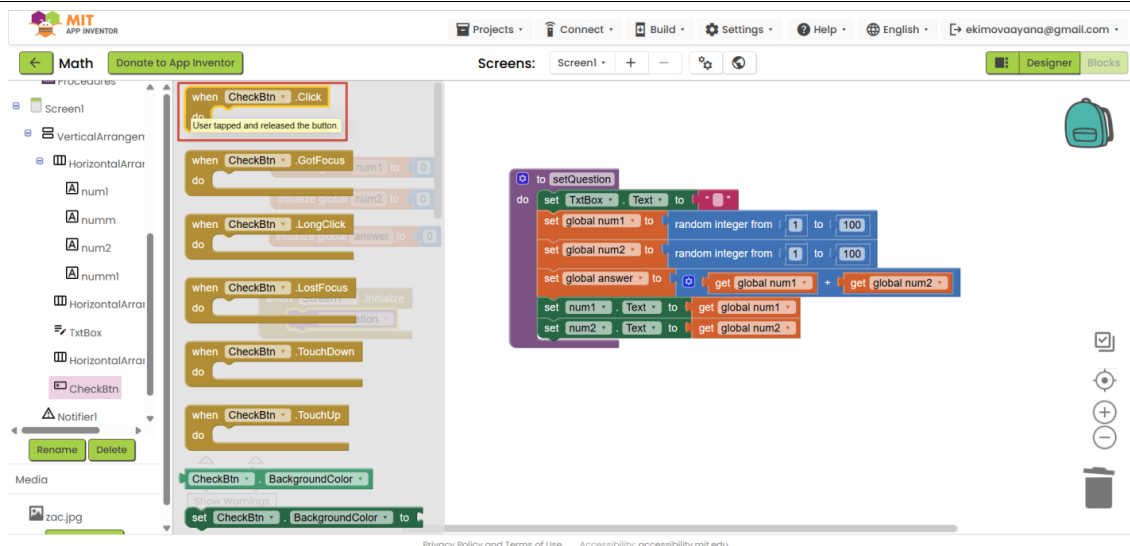


Рисунок 26. Добавление блока события

Дальше добавляем условный блок «if then» из категории Control для реализации логики проверки правильности введенного пользователем ответа, где при истинности условия будет выполняться соответствующий сценарий реакции приложения.

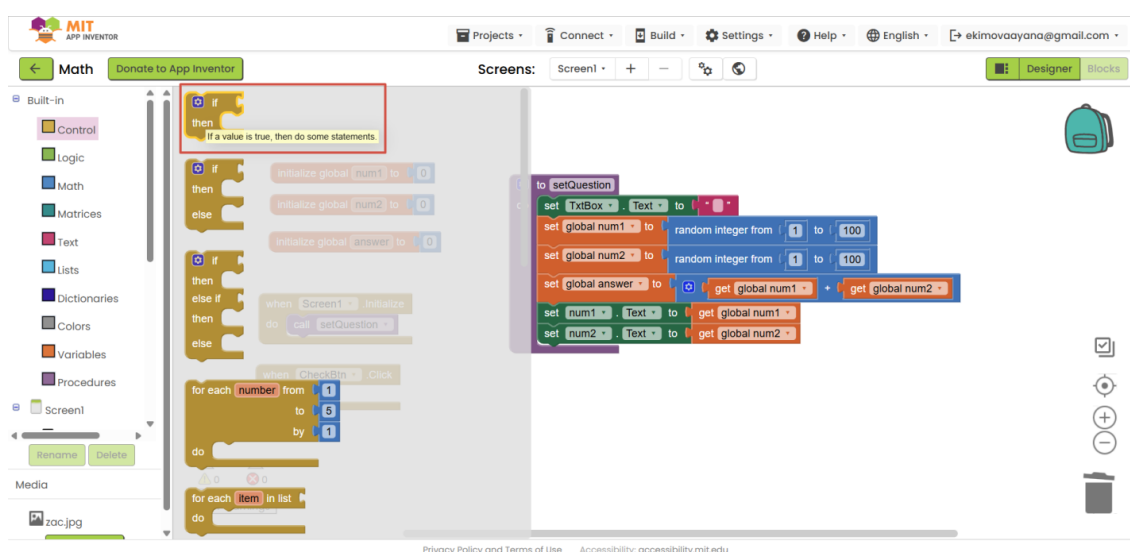


Рисунок 27. Добавление условного блока

Затем добавляем блок сравнения на равенство (=) из категории Math, который необходим для реализации условия проверки: сопоставления введенного пользователем ответа со значением переменной «answer», содержащей правильный результат вычислений.

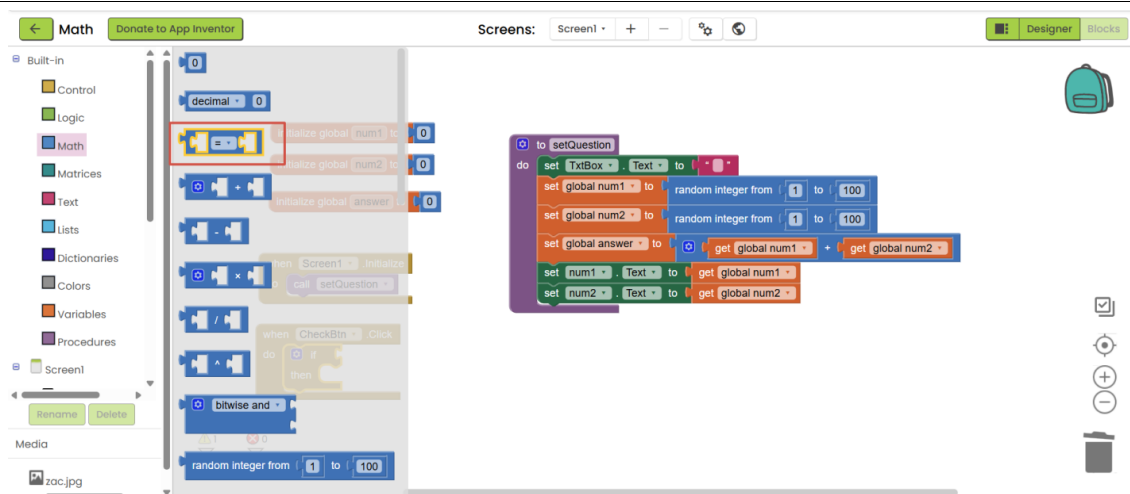


Рисунок 28. Добавление блока сравнения

Вставляем блок сравнения на равенство в условие оператора if, подготавливая структуру для проверки совпадения введенного пользователем ответа с правильным результатом.

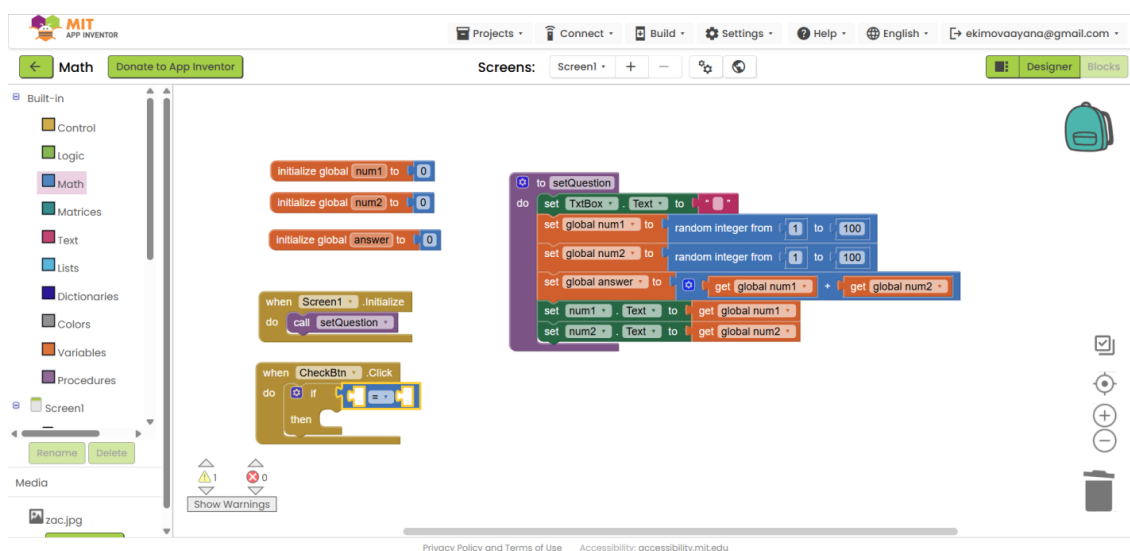


Рисунок 29. Добавление блока сравнения на равенство

Добавляем блок получения текстового содержимого компонента TextBox, который считывает значение, введенное пользователем в поле ответа, для дальнейшего сравнения с вычисленным результатом.

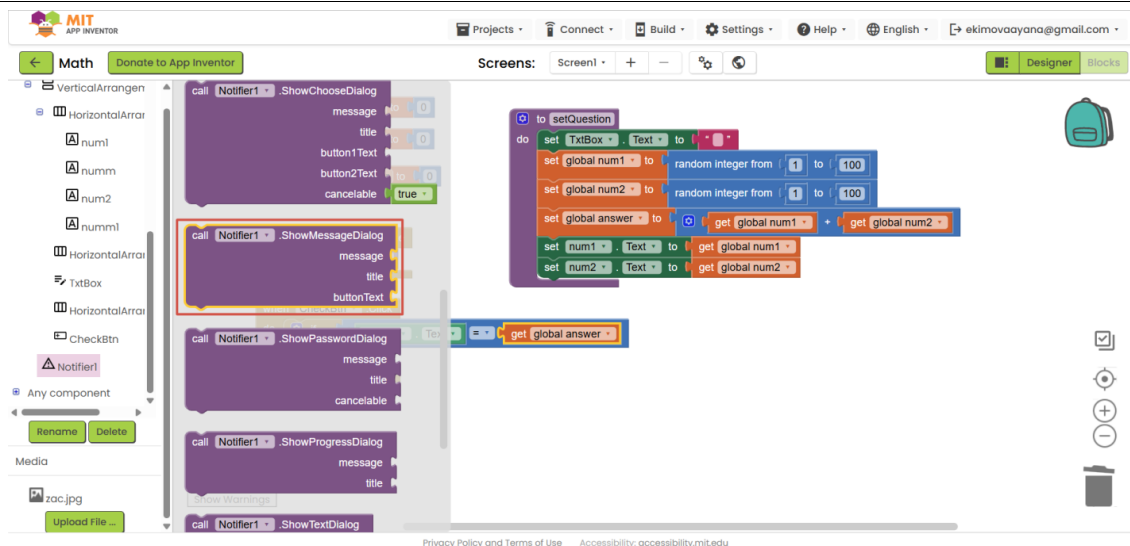


Рисунок 32. Добавление блока вызова

Вставляем блок вызова метода ShowAlertDialog компонента Notifier1 внутри ветки then условного оператора, обеспечивая отображение всплывающего окна с сообщением об успехе при совпадении введенного пользователем ответа с правильным результатом.

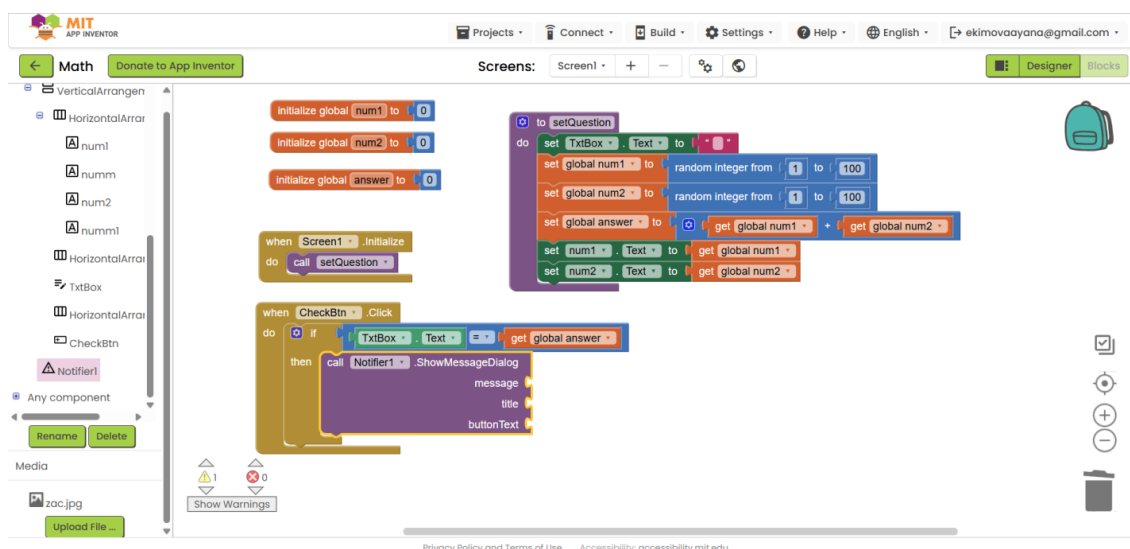


Рисунок 33. Редактирование блока вызова

Заполняем поля текстовых свойств в блоке «ShowAlertDialog», задавая сообщение «Правильно!», заголовок «Хорошая работа!» и текст кнопки «Далее», чтобы предоставить пользователю визуальную обратную связь при верном ответе.

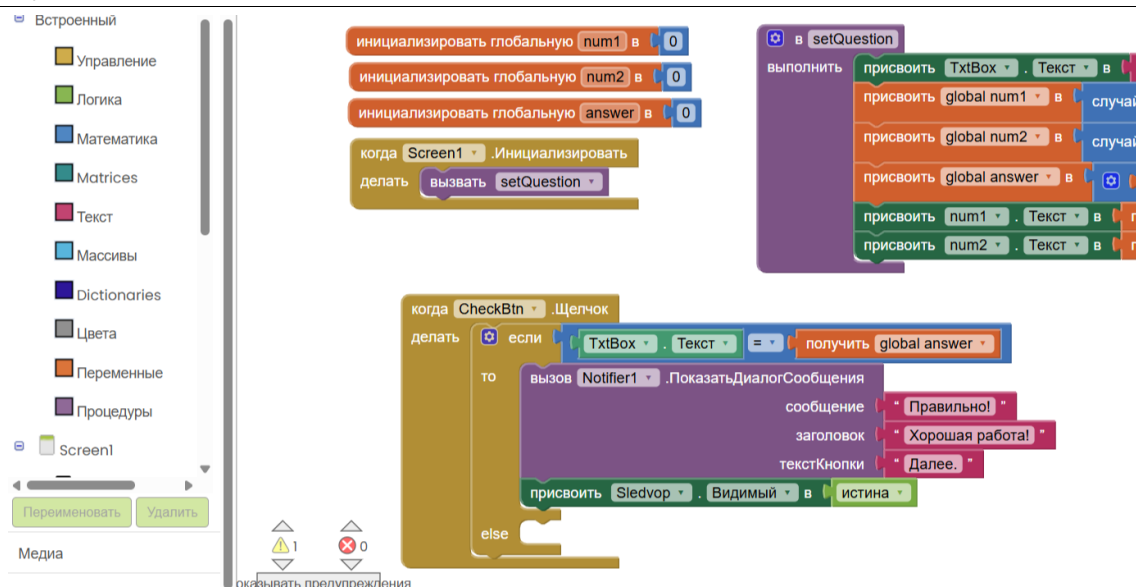


Рисунок 34. Заполнение текстовых полей

Копируем и вставляем ниже блок вызова диалогового окна в ветвь «else», изменяя параметры сообщения на «Неправильно!» и «Попробуй ещё раз!», а также устанавливаем видимость кнопки «Sledvor» в значение «ложь», чтобы запретить переход к следующему вопросу при неверном ответе.

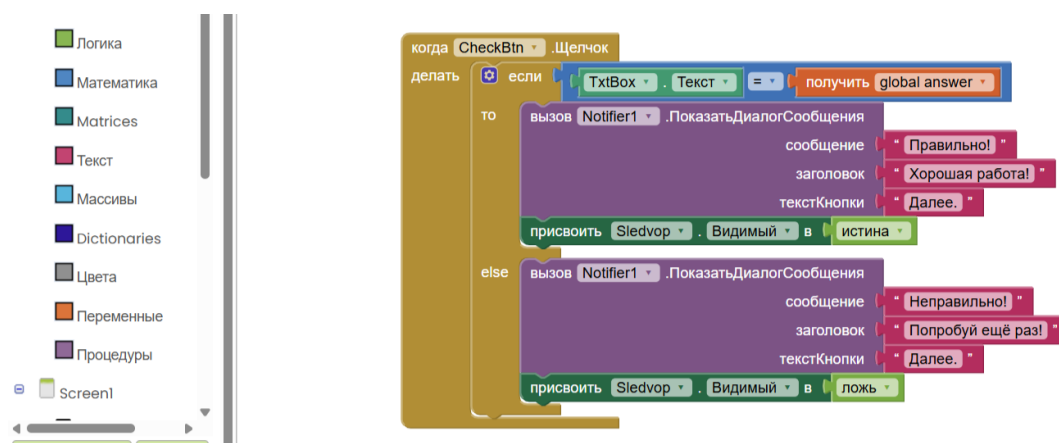


Рисунок 35. Редактирование блока вызова

Копируем и вставляем ниже блок обработчика нажатия кнопки «Sledvor», в котором указываем переход на экран «Screen11», тем самым реализуя функцию перезапуска игры или перехода к следующему вопросу.

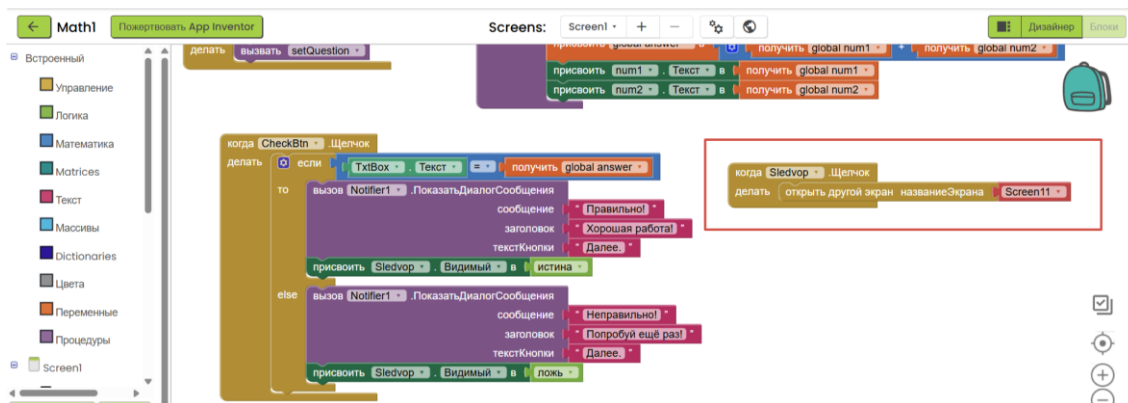


Рисунок 36. Редактирование блока обработчика события

В результате разработки было создано функциональное мобильное приложение с интуитивно понятным пользовательским интерфейсом, на котором отображается сгенерированный математический пример ($77+54=$), поле для ввода ответа и кнопка проверки правильности решения, что демонстрирует успешную реализацию всех запланированных образовательных и игровых механик приложения.

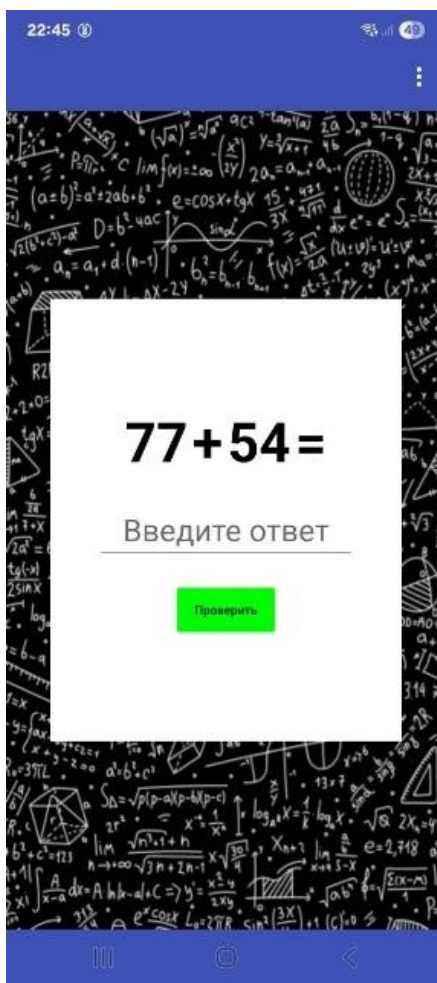


Рисунок 37. Экран 1 с примером

При вводе неверного ответа приложение отображает модальное окно с сообщением «Неправильно! Попробуй ещё раз!», что реализует механизм

интерактивной обратной связи, позволяющий пользователю осознать ошибку и повторить попытку решения, при этом переход к следующему вопросу блокируется до получения верного результата.

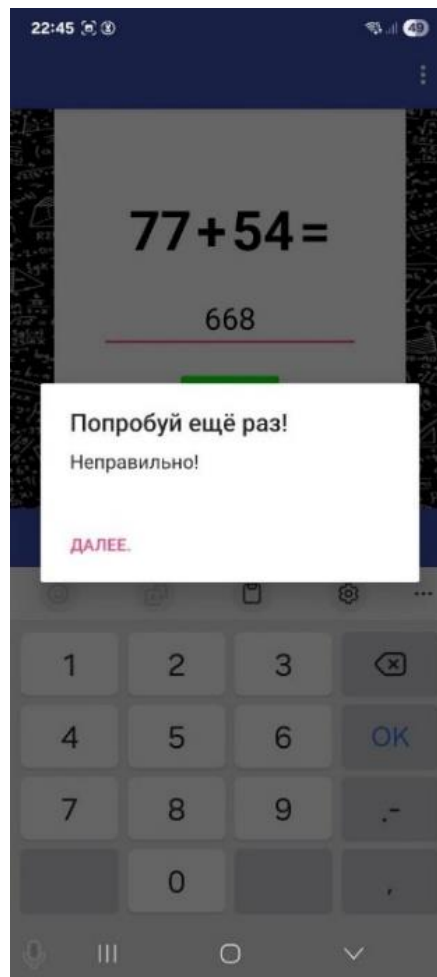


Рисунок 38. Модальное окно с ошибкой

При правильном вводе ответа пользователю выводится модальное окно с сообщением «Хорошая работа! Правильно!», что обеспечивает положительное подкрепление и мотивирует продолжить решение задач, а кнопка «Далее» позволяет сгенерировать новый пример.

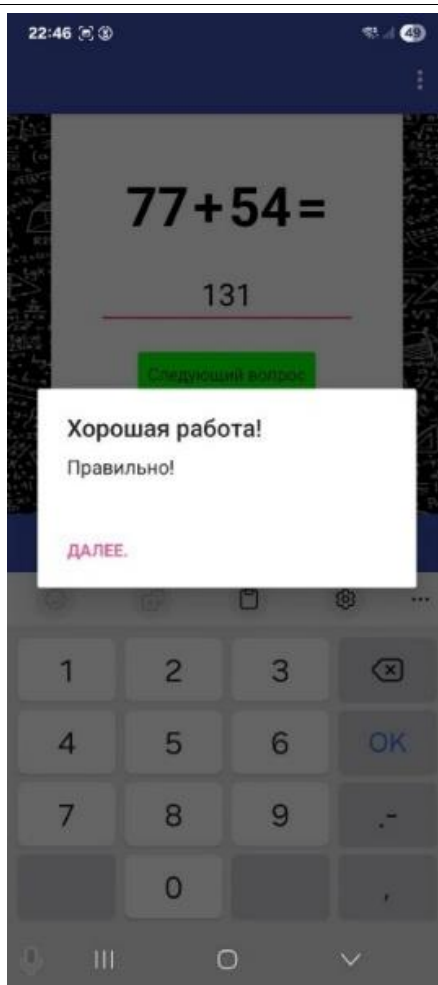


Рисунок 39. Модальное окно с верным ответом

При вводе верного ответа приложение отображает кнопку «Следующий вопрос», что позволяет пользователю перейти к генерации нового примера и продолжить игровой процесс.



Рисунок 40. Переход к следующему вопросу

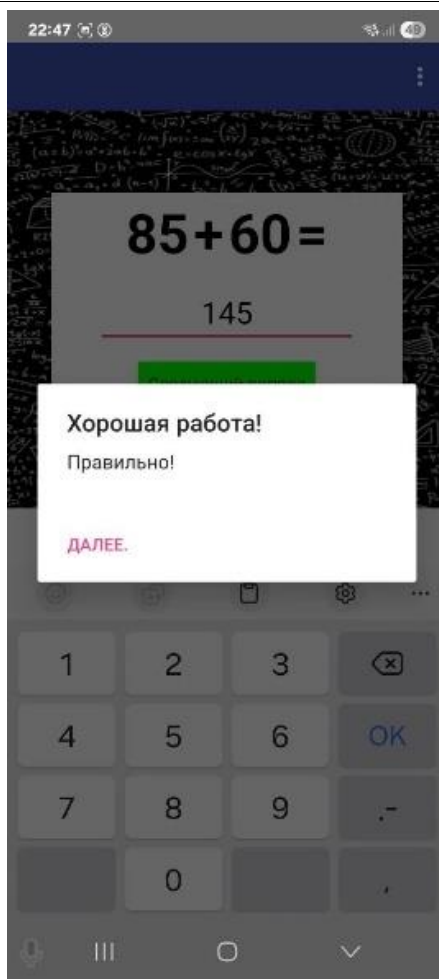


Рисунок 41. Пример 2

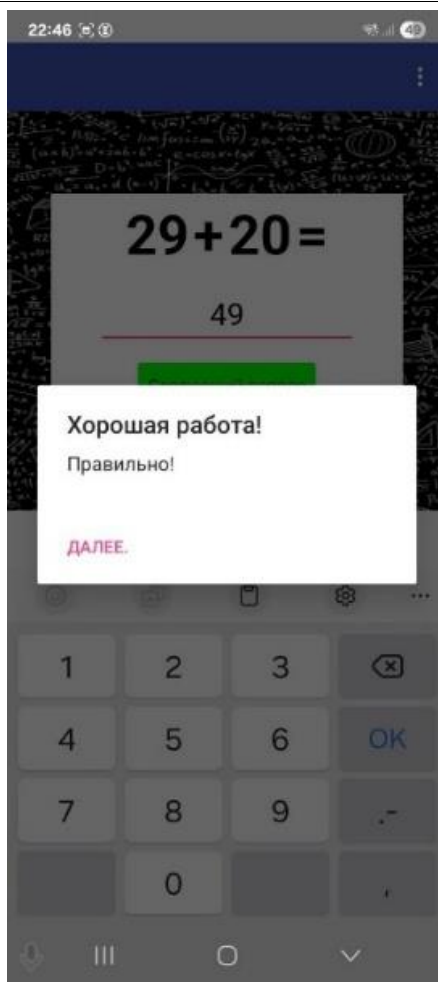


Рисунок 42. Пример 3

Для получения готового приложения выбираем в меню пункт «Построить» и опцию «Android App (.apk)», что инициирует процесс компиляции и генерации QR-кода для загрузки установочного файла на мобильное устройство.

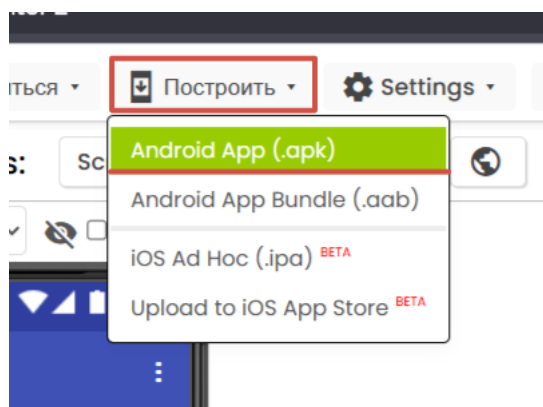


Рисунок 43. Процесс скачивания файла

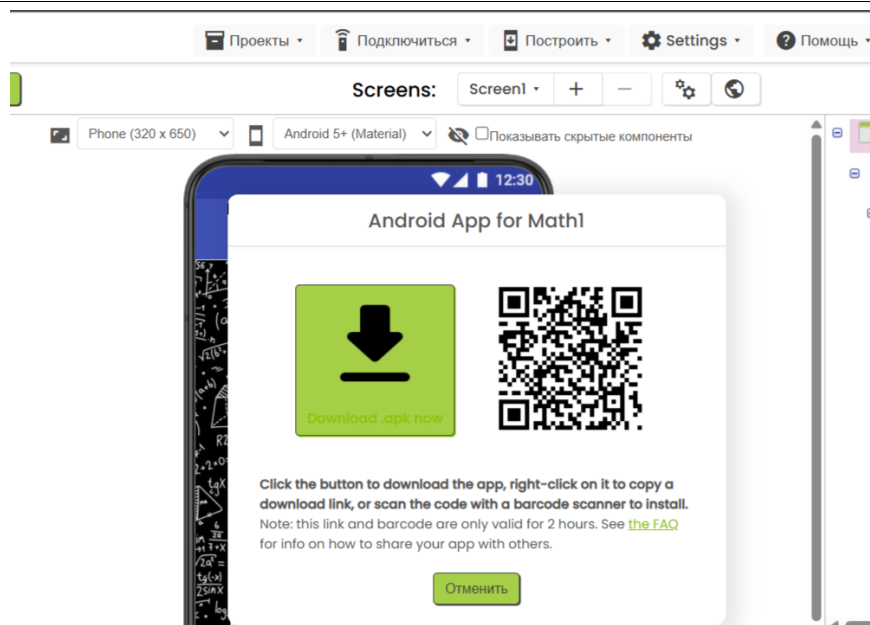


Рисунок 44. Выбор варианта скачивания файла

4 Выводы

В данной работе представлен процесс реализации программной логики математической игры «Math» в среде визуального блочного программирования MIT App Inventor. Использование функциональных возможностей редактора Blocks MIT App Inventor позволило создать полноценное интерактивное приложение с замкнутым игровым циклом: генерация примера — ввод ответа — проверка — обратная связь — переход к новому заданию. Разработанное приложение демонстрирует потенциал платформы для создания функциональных обучающих игр, направленных на формирование вычислительных навыков, и подтверждает эффективность визуального блочного программирования как доступного средства разработки образовательных мобильных приложений.

Библиографический список

1. Тасуов Б., Муратова Г., Сабирова Р. Использование метода проектов на уроках информатики. Создание мобильного приложения в среде mit app inventor //Научно-педагогический журнал «Білім-Образование» Национальной академии образования имени И. Алтынсарина. 2023. Т. 104. №. 1.
2. Самылкина Н. Н., Етова А. В. Создание мобильных приложений в MIT App Inventor с использованием элементов геймификации при изучении программирования в основной школе //Информатика в школе. 2020. №. 8. С. 37-4
3. Вильданов А. Н. Разработка мобильных игр в жанре "Поиск предметов" с помощью MIT App Inventor //Инженерный вестник Дона. 2019. №. 8 (59). С. 7.
4. Малова А. И. Использование визуальной среды App Inventor при обучении

программированию школьников //XIX Всероссийская научно-практическая конференция молодых исследователей образования. – Томский государственный университет, 2020. С. 414.