В. Г. Рындак
В. О. Дженжер
Л. В. Денисова

Проектная деятельность школьника в среде программирования Scratch

Учебно-методическое пособие

Оренбург — 2009

УДК 373.31 (075)
ББК 74.263.2я73
Р 95

Исследования выполнены при поддержке Федерального агентства по образованию в рамках реализации аналитической ведомственной целевой программы «Развитие научного потенциала высшей школы» (2009–2010 годы) (№ 3.1.2/4125).

Рындак В. Г., Дженжер В. О., Денисова Л. В.

Р 95 Проектная деятельность школьника в среде программирования Scratch: учебно-методическое пособие / В. Г. Рындак, В. О. Дженжер, Л. В. Денисова. — Оренбург: Оренб. гос. ин-т. менеджмента, 2009. — 116 с.: ил.

Пособие включает рекомендации по организации проектной деятельности в среде программирования Scratch. Представлены разработки занятий, охватывающих 26 тем познавательного и творческого направления. Предложено примерное тематическое планирование внеучебной проектной научно-познавательной деятельности младшего школьника в соответствии с общеобразовательным стандартом второго поколения. Содержатся примерные темы междисциплинарных проектов для начальной, основной и старшей школы.

Книга может быть полезна учителям начальных классов, информатики, работникам учреждений дополнительного образования, студентам педагогических вузов, родителям и всем, кто заинтересован в интеллектуальном и творческом развитии детей.

УДК 373.31 (075)
ББК 74.263.2я73

Содержание

Введение

Мы живём в эпоху, когда знание перестало быть чем-то единым. Разделённое на «науки» и «дисциплины», оно теряет стройную целостность и предстаёт перед нами в виде лоскутов, из которых скроена картина мира. Школьное (да и профессиональное) образование есть не что иное, как изучение отдельных фрагментов этого пёстрого витража. Можно ли сегодня считаться образованным человеком, не замечая и не понимая связей между элементами реальности? Вероятно, ответ на этот вопрос будет отрицательным. Многие исследователи полагают, что корни проблемы тянутся из организации школьного обучения. Например, Алан Кей[1], который интересуется не только «высокой» наукой, но внимательно следит за обучением детей, считает, что нужно как можно раньше дать ребёнку мощный «инструмент для думания». Основное назначение этого инструмента — познание нового и создание связей между известным, развитие не только аналитического, но и синтетического мышления.

Психологическая наука говорит, что возраст младшего школьника приходится на сензитивный период, когда он способен сознательно осуществлять частично-поисковую деятельность. Это хорошо сочетается с использованием метода проектов, который особенно эффективен при внеурочной форме обучения и способствует усвоению знаний путём разрешения проблемных ситуаций.

В последние годы очень популярным стал язык (и среда) программирования Scratch (читается `Скрэтч`). Это не просто оборот речи: Scratch располагается на 24 месте среди всех (!) самых популярных языков программирования, включая такие, как C, Java, C++, C#, PHP, Pascal и другие[2]. Двадцать четвёртое место — у «игрушечного» детского языка (кстати, у Logo — 32). Это можно объяснить только огромной потребностью и педагогического сообщества в целом, и самих детей в средстве для «думания», исследования и самовыра-

[1] Один из разработчиков графического оконного интерфейса, создатель

[2] См. http://www.tiobe.com/index.php/content/paperinfo/tpci/

жения. К сожалению, Россия не входит в число стран, активно интересующихся Scratch. Мы собираемся, по возможности, исправить эту ситуацию.

Scratch был создан на языке Squeak[3], который сам заслуживает особого разговора. Главным идеологом Scratch является ученик Пейперта Мич Резник из MIT Media Lab (Массачусетский технологический институт; это тот самый университет, где в 1968 г. С. Пейперт разработал Logo). Возможно, язык распространился настолько быстро потому, что создавался не для изучения программирования. Scratch, с точки зрения теории, — объектно-ориентированный язык с возможностью создавать многопоточные программы. С практической точки зрения это простой в изучении, красивый, мощный инструмент, который не требует двухмесячного изучения, прежде чем появится возможность написать программу для решения квадратного уравнения. Никаких двух месяцев и, если не хотите, никаких квадратных уравнений. Писать можно сразу, через десять минут знакомства с основами. Причём в отличие от Бейсика, на котором начать писать тоже очень легко, Scratch не поощряет плохого стиля программирования. Напротив, «правильные» программы на Scratch писать проще, чем «неправильные». Графика, анимация, музыка, видеоэффекты, и в то же время классическое событийно управляемое объектно-ориентированное и модное параллельное программирование. Каждый может брать всё, что ему нужно сейчас. А потом двигаться дальше.

Эта книга была задумана в связи с разработкой общеобразовательных стандартов второго поколения. Как известно, в настоящее время продолжается активная работа по созданию новой образовательной системы в России. Одним из важнейших направлений здесь является разработка и внедрение новых образовательных стандартов. В структуре основных общеобразовательных программ и результатах их освоения проект концепции федеральных государственных образовательных стандартов общего образования [4] выделяет четыре компонента: фундаментальное ядро (определяет

[3] Разновидность Smalltalk.

содержание учебных программ и организацию образовательной деятельности по отдельным учебным предметам), базисный учебный план (регулирует педагогический процесс через инвариантную, вариативную части и внеурочную деятельность учащихся), примерные (базисные) учебные программы по предметам (дополняются программами развития универсальных учебных действий), систему оценки достижения требований стандарта.

Значимым, с нашей точки зрения, является введение в стандарт обязательной внеучебной деятельности учащихся, которая, по замыслу разработчиков, призвана в полной мере реализовать требования стандартов общего образования. Предполагается, что часы, отводимые на внеучебную деятельность, будут использоваться по желанию учащихся и, в то же время, будут являться неотъемлемой частью образовательного процесса в школе. Именно здесь (по крайней мере, вначале) нам видится возможность использования Scratch. Учителя информатики, начальных классов, предметники, родители — все, как мы надеемся, смогут использовать Scratch для организации и проведения своих занятий. Моделирование, презентации, средство для активизации мышления, учебные пособия, межпредметные проекты — вот неполный перечень того, где можно использовать Scratch. Хотя материал подобран для изучения в начальной школе (со второго класса), его можно использовать в среднем и старшем звене, внеся некоторые очевидные поправки. Некоторые темы междисциплинарных проектов для основной и старшей школы приведены в нашем пособии.

Все проекты, о которых идёт речь в пособии можно скачать одним архивом с сайта http://sites.google.com/site/OrenScratch.

Эту книгу мы написали для учителей и родителей, которые, несмотря на повседневные заботы, усталость и нехватку времени, всё-таки не перестали замечать прекрасное, удивляться и радоваться ему.

Уважаемые коллеги! Эта книга — для вас.

1. Общие рекомендации

1.1. Авторская концепция методики организации внеучебной проектной научно-познавательной деятельности школьника с использованием среды программирования Scratch

Ключевым в нашей концепции является понятие «проектная научно-познавательная деятельность школьника», как совместная (с другими субъектами) или самостоятельная деятельность с использованием методов научного исследования, ведущим мотивом которой является познавательный интерес, и организованную в форме выполнения проектов. К методам научного исследования мы относим, например, работу с материалами по данной тематике, выдвижение и опровержение гипотез, планирование и проведение эксперимента, анализ и синтез, публичная защита полученных результатов.

Проектная научно-познавательная деятельность не является самоцелью, но рассматривается нами как среда, в которой наиболее естественным образом раскрывается личностный потенциал школьника. В этой связи ***целями проектной научно-познавательной деятельности*** школьника мы полагаем:

1. развитие интеллектуальных, познавательных и творческих способностей школьника;
2. развитие метапредметных умений (личностных, познавательных, коммуникативных, регулятивных);
3. развитие способов мыслительной деятельности;
4. формирование целостной картины мира и системного мышления на основе межпредметных связей.

Следует иметь в виду, что возрастные особенности младшего школьника не позволяют в полной мере реализовать проведение полноценных научных исследований. В то же время раннее включение в организованную специальным об-

разом проектную деятельность творческого характера позволяет сформировать у школьника познавательный интерес и исследовательские навыки, которые в старшем возрасте пригодятся им для выполнения научно-познавательных проектов.

Организация научно-познавательной деятельности школьника требует использования инструмента (средства) для выполнения как исследовательских, так и творческих проектов. В качестве такого инструмента мы видим среду программирования Scratch (http://scratch.mit.edu). Наш выбор обусловлен следующими соображениями.

Во-первых, программная среда должна быть легка в освоении и понятна даже ученику начальной школы, но, в то же время, должна давать принципиальную возможность составлять сложные программы. Это позволяет постепенно направлять деятельность школьника в русло научно-познавательного исследования, не расходуя при этом силы на изучение каждый раз новой программной среды.

Во-вторых, нужная нам среда должна позволять заниматься как программированием, так и созданием творческих проектов. Это позволит вовлечь во внеучебную деятельность ребят не только с абстрактно-логическим, но и с преобладающим наглядно-образным мышлением.

Наконец, в-третьих, нам нужен программный инструмент, вокруг которого сложилось активно действующее, творческое, разнонаправленное, позитивно настроенное интернет-сообщество. Мы предполагаем, что школьники будут использовать его как пространство идей, как референтную группу для собственных проектов, как стимул для созидания.

Существует значительное количество разных учебных программных средств: языки высокого уровня (Pascal, C, C++, C#, Java и др.), специальные учебные языки (Logo, PervoLogo, ЛогоМиры, Star Logo, NetLogo, Squeak, Green Foot на основе Java и др.), разнообразные и очень популярные школьные исполнители (Роботландия, Кенгурёнок, Чертёжник, Кузнечик, Паркетчик и др.). Но ни одно из этих средств не удовлетворяет в полной мере перечисленным требованиям. Мы остановили свой выбор на среде Scratch после анализа её педагогического потенциала.

1.2. Потенциал среды программирования Scratch в организации проектной научно-познавательной деятельности школьника

Среда программирования. Scratch — это, в первую очередь, система программирования, обладающая всеми необходимыми атрибутами. Scratch имеет собственный редактор текста программы, построенный на интересной идее конструкторов Lego: все операторы языка и другие его элементы представлены блоками, которые могут соединяться один с другим, образуя скрипт (фрагмент кода). Важной особенностью блоков является их «специализация»: имеется несколько видов блоков и они могут составляться не произвольным образом, а лишь сообразно своему назначению. Так и в конструкторе Lego не каждую деталь можно соединить с любой другой. Это ограничивает количество возможных вариантов соединения, и, соответственно, исключает возможность появления синтаксических ошибок. Кроме этого имеется транслятор, унаследованный от Squeak, и отладчик, позволяющий выполнять программы в пошаговом режиме.

Как язык программирования, Scratch представляет собой разновидность объектно-ориентированного языка, наследника первого объектно-ориентированного языка Smalltalk. Такая родословная Scratch позволяет программисту свободно использовать его в качестве инструмента для моделирования объектов и процессов реального мира. Встроенная и интуитивно понятная графическая подсистема языка позволяет легко проводить визуализацию динамики модели, а также включать в неё элемент интерактивности.

Одной из важнейших особенностей Scratch как языка программирования является его событийно-ориентированный характер. Это означает, что все объекты взаимодействуют при помощи обмена сообщениями. Такая схема обмена информацией делает Scratch близким к современным объектно-ориентированным языкам и позволяет впоследствии более просто организовать переход к изучению Java, Delphi, C# и др.

Scratch является языком, в котором последовательно реализована идея многопоточности. Каждый скрипт любого объекта запускается в отдельном потоке. В отличие от процедурных языков, в которых принято последовательное выполнение кода, в Scratch скрипты могут выполняться и параллельно. Причём никаких дополнительных действий выполнять не требуется: всё это особенности языка.

Наконец, мы не знаем другого языка программирования, который бы имел средства для написания программ, умеющих получать информацию извне (из реального мира) без помощи клавиатуры и мыши. При наличии специального устройства, подключаемого к порту USB, программа на Scratch может «слышать» звуки, «чувствовать» освещённость в комнате, «ощущать» движения пользователя и выполнять в соответствии с полученной информацией различные действия.

Таким образом, можно выделить следующие имманентные ***свойства Scratch***, имеющие значительный педагогический потенциал.

Простота и дружественность интерфейса позволяют начинать изучение программирования, как только дети научатся читать.

Редактор текстов как конструктор дает возможность на подсознательном уровне превратить «учёбу» в «не учёбу», а кроме того — сократить количество ошибок в программе.

Ориентированность на графику, так как доказана эффективность обучения с опорой на наглядно-образное мышление.

Объектная ориентированность позволяет изучить основные способы создания программ с объектами. Заметим, что Scratch — своеобразный объектно-ориентированный язык. В нём отсутствуют концепции, без которых многие вообще не представляют себе объектно-ориентированную технологию. Например, при изучении Scratch ученик не описывает классов, не использует полиморфизм, не строит деревьев наследования, не сталкивается с инкапсуляцией. Есть объект, его поля и методы (переменные и скрипты). Но отсутствие названных

концепций создаёт *трудности* программирования, особенно заметные при разработке сложных проектов. Многочисленные повторения кода ведут, в конце концов, к идее наследования, многочисленные однотипные сообщения для разных объектов — к идеям полиморфизма и абстрактных классов. Лёгкость изменения полей (и совершения ошибок при таких изменениях) приводит к мысли о сокрытии данных, то есть инкапсуляции. При этом важно, что эти идеи не сообщаются педагогом неподготовленному ученику (даже для студентов 1-2 курса физико-математического факультета названные концепции бывают слишком абстрактны), а самозарождаются, и впоследствии падают на подготовленную почву.

Ориентация на обработку событий способствует формированию метаумения устанавливать причинно-следственные связи, развитию логического мышления, закладывает основы системного мировосприятия через демонстрацию систем как объектов со связями. Наблюдая за поведением таких объектов, ученик приобретает умение отделять существенные признаки предмета от несущественных. Такая аналитико-синтетическая работа характерна для начальных этапов поисково-исследовательской деятельности (Н. А. Менчинская).

Многопоточность позволяет не просто строить модели объектов, но создавать модели действительно комплексных систем, причём без излишних технических сложностей. Кроме того, параллельное программирование сегодня является чрезвычайно востребованной технологией, что делает раннее знакомство детей с ней полезным в плане будущей профессиональной ориентации.

Среда проектирования. Scratch не только язык программирования, но и удачная среда для проектной деятельности, поскольку всё необходимое для такой деятельности включено в его состав. Кроме названных выше редактора, компилятора и отладчика имеются:

– графический редактор для создания и модификации визуальных объектов;

– библиотека готовых графических объектов (некоторые из них содержат наборы скриптов);

– библиотека звуков и музыкальных фрагментов;
– большое количество примеров.

Непосредственно в среду Scratch не входит, но также играет огромную роль динамичное и дружелюбное *сообщество любителей Scratch*, к которому можно подключиться при помощи Интернета. В сообществе, организованном при Массачусетском технологическом институте (США), можно разместить свои работы, послушать похвалы и критику, найти единомышленников, почерпнуть новые идеи. Одна из ключевых особенностей Scratch состоит в том, что проекты распространяются с исходными кодами, так что в принципе отсутствует (или почти отсутствует) идея копирайта. Поскольку речь не идёт о коммерческих разработках, это не вызывает недовольства. Напротив, именно благодаря такой свободе возможен быстрый обмен идеями, что не может не сказаться на количестве работ, выложенных в открытый доступ.

Несмотря на то, что каждый может взять любую работу и позаимствовать оттуда код, повторяющихся работ (попросту говоря, — плагиата) очень мало. И это несмотря на то, что заимствования кода приветствуются. По всей видимости, здесь мы наблюдаем процесс самоорганизации, в результате которого банальный плагиат трансформируется в заимствование и *дальнейшее развитие идей*.

Уточним возможность реализации **междисциплинарных проектов**. Именно междисциплинарность позволит школьнику создать единую картину мира, наводя мостики между различными, иногда, на первый взгляд, довольно далёкими друг от друга науками. В этом случае возможности Scratch неоценимы, так как он доступен не только представителю точных наук. Учитель-гуманитарий, например, может использовать Scratch для создания *динамичных и интерактивных презентаций*.

Отметим, что работа в среде Scratch может быть организована как индивидуально, так и коллективно. Разделение функций и ролей среди участников проекта может быть основано на следующих принципах:

– *по функции или роду деятельности (сценарист, художник, программист и т. п.);*

– *по частям проекта (каждый участник выполняет одновременно несколько ролей, разрабатывая свою часть общего проекта).*

Первый способ деления может быть применён в разновозрастных группах, в группах с разным опытом проектной работы, в группах, где учащиеся имеют различные способности или типы мышления, т. е. в разнородных группах. Второй способ подходит для однородных групп. Однако эти рекомендации не носят жёсткого характера и могут варьироваться в зависимости от ситуации.

Среда моделирования. Перечисленные особенности Scratch делают его очень удобной, практически идеальной средой для обучения моделированию.

Моделирование представляет собой один из наиболее универсальных методов познания действительности. Для нас в настоящем контексте важен его педагогический потенциал, а также возможность реализовать этот потенциал в Scratch.

Мысленный образ модели основан на анализе изучаемого объекта и может быть представлен в примитивах некоторого языка своими компонентами и связями. В нашем случае идеальный объект моделирования разделяется на компоненты, а затем воссоединяется (синтезируется) в новой среде, отличной от мозга, а именно в среде Scratch: происходит объективация идеальной структуры. Такой подход характерен для построения любой модели, поэтому простейшие аналитико-синтетические действия будут вынужденно производиться учеником.

Следующим шагом после создания модели является проверка её на адекватность и, если нужно, — коррекция. Тестирование производится в режиме игры с моделью. Это не наблюдение за столбцами цифр, не слежение за колебаниями абстрактных графиков, а именно игра. Во время игры автор замечает свои недоработки, неточности и ошибки. Затем соотносит их с кодом и вносит необходимые исправления. Заметим, что во многих курсах моделирования этап анализа проводится очень скомкано. Происходит это в силу того, что результаты моделирования слишком абстрактны и не вызывают сопереживания у учеников. С моделями на Scratch ин-

тересно поиграть, а если в «игре» что-то не так, то можно легко исправить ошибку.

Среди моделей на Scratch можно выделить: простую или интерактивную анимацию; феноменологическую модель объекта, процесса или явления; математическую модель.

Уточним, что может быть очень простая программа для реализации математической модели и очень сложная для интерактивной анимации.

Таким образом, от других языков программирования, используемых в школе, Scratch отличается изначальной направленностью на создание моделей, работу с моделями. Это делает его незаменимым инструментом для организации проектной научно-познавательной деятельности.

Среда для творчества. Было бы неверным представлять Scratch только как средство организации учебной деятельности. Он может использоваться и как инструмент творчества. В Интернете огромное количество проектов исключительно эстетической направленности. Огромное количество визуальных эффектов делает Scratch очень привлекательным в качестве средства самовыражения. Несмотря на отсутствие «научности», такие проекты лишь первый шаг к более серьёзной проектной научно-познавательной деятельности учеников, так как по мере роста мастерства растут и здоровые амбиции создателей. Хочется новых идей, новых инструментов, что и выводит автора «творческих» проектов к «исследовательским» проектам (слова здесь взяты в кавычки потому, что зачастую очень сложно бывает провести границу между проектами разного вида).

Отметим, что Scratch может использоваться как единый инструмент для самых различных возрастов и типов мышления. Практически с того момента, как ребёнок научился читать (и даже раньше: просто в этом случае блоки языка рассматриваются как своеобразные иероглифы) и до 14-16 лет.

Обобщая сказанное, выделим педагогический потенциал как совокупность ресурсов, возможностей и способностей среды Scratch.

Под **ресурсами** Scratch мы понимаем все его особенности как языка и системы программирования. В первую очередь к ним относятся: объектная ориентированность; поддержка событийно-ориентированного программирования; параллельность выполнения скриптов; дружественный интерфейс; разумное сочетание абстракции и наглядности; организация текстов программ из элементарных блоков; наличие средств взаимодействия программ на Scratch с реальным миром посредством дополнительного устройства; встроенная библиотека объектов; встроенный графический редактор; активное интернет-сообщество пользователей.

К **возможностям** Scratch отнесём проекцию его ресурсов в психолого-педагогический и методический планы, то есть те его свойства, которые напрямую проистекают из наличных ресурсов. Наиболее существенны, на наш взгляд, возможности Scratch направленные на: изучение основ алгоритмизации; изучение объектно-ориентированного и событийного программирования; знакомство с технологиями параллельного программирования; моделирование объектов, процессов и явлений; организацию проектной деятельности, как единоличной, так и групповой; организацию научно-познавательной деятельности; установление межпредметных связей в процессе проектной и научно-познавательной деятельности; организацию кружковой работы с направленностью на художественное творчество.

Способности Scratch определяются нами как проявление его возможностей в отношении развития личностных качеств учеников. Потенциальность этой связи заключается в вероятностном характере объективации возможностей Scratch. К наиболее значимым новообразованиям относятся [8]: ответственность и адаптивность; коммуникативные умения; творчество и любознательность; критическое и системное мышление; умения работать с информацией и медиасредствами; межличностное взаимодействие и сотрудничество; умения ставить и решать проблемы; направленность на саморазвитие; социальная ответственность.

Таким образом, педагогический потенциал среды программирования Scratch позволяет рассматривать её как

перспективный инструмент (способ) организации междисциплинарной внеучебной проектной научно-познавательной деятельности школьника, направленной на его личностное и творческое развитие.

И, наконец, перечисленные особенности Scratch оказывают влияние на развитие таких личностных качеств ученика [8]: ответственность и адаптивность; коммуникативные умения; творчество и любознательность; критическое и системное мышление; умения работать с информацией и медиасредствами; межличностное взаимодействие и сотрудничество; умения ставить и решать проблемы; направленность на саморазвитие; социальная ответственность.

Таким образом, в качестве способов организации внеучебной проектной научно-познавательной деятельности школьника мы выделяем:

1. использование среды программирования Scratch в качестве системообразующего элемента;
2. выполнение научно-познавательных и творческих проектов междисциплинарного характера;
3. работа над выполнением проектов в разновозрастных группах.

К наиболее существенным *особенностям* предлагаемой модели внеучебной деятельности мы относим:

1. выполнение проектов в среде программирования Scratch (с возможностью впоследствии перейти к другим средам);
2. возможность как индивидуальной, так и групповой работы (в том числе в разновозрастных группах);
3. работу на выбранном уровне сложности;
4. отсутствие жёсткого регламента, что предполагает возможную необязательность посещения занятий, выполнения заданий и т. п., т. е. индивидуальную образовательную траекторию для каждого ученика;
5. безотметочная система оценивания;
6. свободный выбор тематики работы при поощряемой полидисциплинарности;

7. доведение проекта до защиты как одно из наиболее важных правил;
8. возможность свободно обмениваться мнениями, как внутри своей группы, так и с коллегами;
9. равноправие «научных» и «творческих» проектов.

Выделим некоторые *типы проектов*, выполняемых в среде Scratch:

- музыкальный проект;
- анимация;
- комикс;
- интерактивная игра;
- графика;
- с элементами искусственного интеллекта (ИИ);
- учебная презентация;
- учебная модель, демонстрационный эксперимент;
- обучающая программа.

По содержанию проекты разделяются на свободные и предметные (например, в рамках школьной дисциплины).

Для успешной работы в среде Scratch желательно, чтобы школьник имел предварительную практику работы за компьютером, включающую знания и умения и по следующим направлениям:

- назначение основных устройств компьютера для ввода и вывода информации;
- включение и выключение компьютера и подключаемых к нему устройств;
- операционная система MS Windows:
  - рабочий стол;
  - панель задач (назначение каждой из четырёх функциональных частей: кнопка «Пуск», панель быстрого запуска приложений, область задач, системная панель);
  - значки;
  - файлы и папки;
  - имя файла;

- размер файла;
- сменные носители;
- адрес файла;
- операции над файлами и папками (создание, сохранение, открытие, копирование, переименование);
- навыки набора текста на компьютере;
- техника работы с мышью;
- окна и их элементы (меню, панели инструментов, вкладки, кнопки, полосы прокрутки, выпадающие списки и др.);
- запуск и завершение программы.

К общеучебным умениям, которыми должен владеть школьник для успешного вхождения в данный курс, мы относим умение читать.

Как видно, указанные требования к специальным (связанным с информационными технологиями) и общеучебным умениям являются минимальными и позволяют проводить занятия в среде Scratch уже в начальной школе. Тем более что проектно-преобразовательная деятельность младшего школьника тесно связана с коммуникативной, игровой и эстетической и по существу носит творческий характер [12].

1.3. Требования к уровню подготовки выпускника начальной школы

Среди требований к уровню подготовки выпускника начальной школы мы выделяем приёмы проектной деятельности и освоенность средства проектной деятельности — среды Scratch.

Требования к уровню усвоения приёмов проектной деятельности

Младший школьник, участвующий в проектной научно-познавательной деятельности, по окончании начальной школы **должен**:

знать

1. отдельные способы планирования деятельности:

1.1. составление плана предстоящего проекта в виде рисунка, схемы;
1.2. составление плана предстоящего проекта в виде таблицы объектов, их свойств и взаимодействий;
1.3. разбиение задачи на подзадачи;
2. распределение ролей и задач в группе;

уметь

1. составить план проекта, включая:
1.1. выбор темы;
1.2. анализ предметной области;
1.3. разбиение задачи на подзадачи;
2. проанализировать результат и сделать выводы;
3. найти и исправить ошибки;
4. подготовить небольшой отчёт о работе;
5. публично выступить с докладом;
6. наметить дальнейшие пути развития проекта;

иметь первичные навыки

1. работы в группе;
2. ведения спора;
3. донесения своих мыслей до других.

Требования к уровню освоенности средства проектной деятельности — среды программирования Scratch

Младший школьник, участвующий в проектной научно-познавательной деятельности с использованием среды Scratch, по окончании начальной школы **должен**:

знать

1. Алгоритмы и блоки
1.1. Понятие алгоритма
1.2. Исполнитель
1.3. Система команд исполнителя
1.4. Реализация алгоритмов: блоки Scratch
1.4.1. Движение
1.4.2. Контроль
1.4.3. Внешность
1.4.4. Числа
1.4.5. Перо
1.4.6. Звук
1.4.7. Сенсоры

2. События
   2.1. Виды событий
   2.2. Сообщения
      2.2.1. Источник
      2.2.2. Адресат
      2.2.3. Обработчик
3. Графический редактор
   3.1. Рисование
   3.2. Модификация
   3.3. Центрирование
4. Математический базис
   4.1. Отрицательные числа
   4.2. Декартова система координат
   4.3. Десятичные дроби
   4.4. Операции отношения
   4.5. Логические операции «И», «ИЛИ»
   4.6. Случайные числа
   4.7. Арифметические операции и функции
   4.8. Градусная мера угла
5. Объекты
   5.1. Создание
   5.2. Свойства
   5.3. Методы (скрипты)
   5.4. Последовательность и параллельность
   5.5. Взаимодействие

уметь

1. работать в среде Scratch.

1.4. Организация внеучебной проектной деятельности школьника с использованием среды Scratch

В начальной школе (2–4 классы) занятия должны проводиться из расчёта 3 часа в неделю, 34 недели в год в течение трёх лет; итого 306 часов (в соответствии с базисным учебным планом стандартов второго поколения [2]). За компьютером учащиеся проводят только часть этого времени, а остальные часы делятся между изучением литературы, вы-

бором тем, планированием, подготовкой докладов, участием в конференциях и пр.

Предлагаемые учебный план и рабочая программа рассчитаны на 2–3 классы (204 часа с резервом времени в 16 часов), однако могут быть использованы не только в начальном и общем, но и в дополнительном образовании без существенных доработок, т. к. являются «точками входа» во внеучебную проектную научно-познавательную деятельность. Это означает, что начинать обучение по этим планам можно со 2 класса, но практически эти материалы могут быть использованы для работы в более старших классах, если со средой Scratch школьники не знакомы.

Количество часов, отводимых на каждый проект, условно и может варьироваться в зависимости от уровня подготовленности школьников. Кроме того, следует учитывать специфику собственно проектной деятельности, предполагающей движение в собственном темпе и направлении. Возможны ситуации, когда часть ребят справится с заданием раньше остальных. В таком случае оставшееся время они будут заняты свободным проектированием.

Если точка входа в проектную деятельность приходится на второй класс, то овладев основами среды Scratch к четвёртому классу, школьники смогут заниматься собственно межпредметной проектной деятельностью. Аналогично, если точкой входа является пятый класс.

Таблица 1

Примерное тематическое планирование внеучебной проектной деятельности школьника (точка входа)

№	Темы проектной деятельности	Часов
1.	***Анимация***. Знакомимся со Scratch. Наш Кот ходит и мяукает! **Цель**: знакомство со средой Scratch.	3
2.	***Интерактивная анимация.*** Ска́чки. Щекочем Лошадку. **Цель**: закрепление изученного с дополнительным продвижением в изучении интерфейса среды.	3
3.	***Музыкальный***. Играем на пианино и других	2

№	Темы проектной деятельности	Часов
	музыкальных инструментах. **Цель**: знакомство с музыкальными возможностями Scratch.	
4.	***Музыкальный***. Записываем и сочиняем музыку. **Цели**: (1) научиться записывать музыку с нот; (2) развитие творчества.	3
5.	***Анимация с обработкой событий.*** Ска́чки-2. **Цель**: изучение взаимодействия объектов на основе обмена сообщениями.	4
6.	***Анимация***. Используем слои. **Цели**: (1) закрепление изученного материала; (2) научиться перемещать объекты в различные слои.	2
7.	***Свободное проектирование***. Планируем и делаем мультфильмы и комиксы. **Цель**: (1) знакомство с этапами проектирования; (2) развитие творчества.	6
8.	***Анимация.*** Создаём свой объект в графическом редакторе. **Цель**: научиться создавать собственные спрайты и анимировать их.	4
9.	***Анимация.*** Анимируем полёт пчелы. **Цель**: научиться создавать костюмы к готовым объектам папки Costumes.	2
10.	***Музыкальный***. Создаём оркестр (синхронизируем многоголосье). **Цели**: (1) закрепить навыки работы с графическим редактором; (2) закрепить навыки синхронизации скриптов при помощи сообщений; (3) закрепить навыки создания музыкальных композиций.	4
11.	***Анимация.*** Создаём плавные анимации. **Цели**: (1) познакомиться с понятием «система координат» и научиться соотносить движение спрайта с системой координат Scratch; (2) закрепить понятие параллельности потоков.	6

№	Темы проектной деятельности	Часов
12.	***Анимация с элементами ИИ.*** Изменяем Кота в зависимости от окружающих условий. **Цель**: знакомство с командами ветвления.	3
13.	***Свободное проектирование***. Создаём мультфильмы и комиксы. **Цели**: (1) закрепление изученного материала; (2) закрепление этапов планирования; (3) включение в деятельность обсуждения проектов; (4) развитие навыков самоконтроля.	12
14.	***Анимация с элементами ИИ.*** Знакомимся с переменными. **Цели**: (1) познакомиться с задачами, в которых возникает необходимость в переменных; (2) познакомиться с группой блоков *переменные*.	6
15.	***Анимация.*** Разворачиваем Пчелу в направление движения. **Цели**: (1) закрепить понятие переменной; (2) закрепить понятие системы координат.	2
16.	***Свободное проектирование***. Делаем мультфильмы, комиксы, игры. **Цели**: (1) развитие творчества; (2) закрепление планирования в виде составления таблицы объектов, их свойств и взаимодействий; (3) развитие умений коллективной работы (распределение ролей, задач, навыков взаимодействия); (4) развитие чувства ответственности; (5) постепенный переход к более сложным проектам.	12
17.	***Графика***. Изучаем повороты. **Цели**: (1) познакомиться с градусной мерой углов; (2) познакомится с группой блоков *перо* (аналог языка Logo).	3
18.	***Графика***. Создаём своего исполнителя. **Цели**: (1) закрепить понятия градусной меры угла и поворота; (2) вспомнить понятие исполнителя.	3
19.	Графика с элементами ИИ. Изменяем направ-	2

№	Темы проектной деятельности	Часов
	ление движения в зависимости от условия. **Цели**: (1) закрепить понятие градусной меры угла; (2) вспомнить команды ветвления.	
20.	***Графика***. Рисуем разноцветные геометрические фигуры. **Цели**: (1) закрепить понятие градусной меры угла; (2) изучить средства рисования группы *перо*; (3) познакомиться с выражением единиц в процентах; (4) познакомиться с правильными геометрическими фигурами и изучить способы их рисования.	4
21.	***Свободное проектирование***. ***Графика***. Рисуем натюрморт, пейзаж, портрет. **Цели**: (1) развитие творчества; (2) закрепление этапов планирования.	12
22.	***Игра***. Создаём самую настоящую игру. **Цели**: (1) изучение понятия переменной; (2) изучение планирования в виде составления таблицы объектов, их свойств и взаимодействий.	12
23.	***С элементами ИИ***. Кот анализирует сложную окружающую обстановку. **Цель**: изучить логические операции и соответствующие им блоки в разделе *операторы*.	2
24.	***Свободное проектирование***. ***Игра***. Создаём игры. **Цели**: (1) развитие творчества; (2) закрепление планирования в виде составления таблицы объектов, их свойств и взаимодействий; (3) развитие умений коллективной работы (распределение ролей, задач, навыков взаимодействия) и чувства ответственности.	18
25.	***Интерактивный***. Организуем диалог с пользователем. Написать простую программу, ведущую диалог с пользователем от имени Кота. **Цели**: (1) изучить тип данных «строка»; (2)	10

№	Темы проектной деятельности	Часов
	познакомиться с группой строковых блоков в разделах *операторы* и *сенсоры;* (3) научиться использовать строки при создании диалоговых проектов.	
26.	***Свободное проектирование.*** Создаём игры и творческие проекты. **Цели**: (1) развитие творчества; (2) приобретение и развитие умений коллективной работы.	36
27.	***Публичная защита проектов***. **Цели**: (1) развитие коммуникативных умений; (2) развитие умений публичных презентаций результатов деятельности.	12
28.	***Резерв учебного времени.***	16
	ИТОГО:	204

2. Методические рекомендации к внеучебной проектной деятельности школьника с использованием среды Scratch

2.1. Методические рекомендации к проведению занятий

В начальной школе (точка входа) основной упор при организации проектной деятельности школьника делается на изучении среды Scratch, как основного инструмента осваиваемой деятельности. Помимо этого значительное внимание должно уделяться планированию деятельности, как важному этапу проектирования. Поэтому на реализацию каждого, даже самого простого, проекта отводится довольно много часов. Это связано с тем, что для младшего школьника работа над проектом — новая, а потому сложная деятельность. Поэтому следует оказывать школьнику помощь и поддержку и, в то же время, всячески поощрять его самостоятельность.

Отметим одну особенность работы в среде Scratch, характерную для современных программных сред. Так же как в MS Word мы работаем с документом, а в MS Excel — с книгой, так в Scratch мы всегда работаем над проектом. Это связано, по всей видимости, с тем, что создание любого, даже самого простого продукта в Scratch — анимации, мелодии, презентации и т. п., всегда требует наличия вполне определённой цели деятельности, постоянной сверки полученного результата с исходным замыслом и исправления ошибок. При работе над сложным проектом, состоящим из большого количества объектов, которые содержат сложный программный код, возникает необходимость разбиения исходного проекта на подзадачи. При этом решать каждую такую подзадачу в принципе могут разные участники единого проекта.

Как правило, если цель конечная цель проекта содержится у ребёнка в голове (а не представлена в виде рисунка, таблицы или плана), возможны многократные и трудно исправимые ошибки при реализации проекта. В этом случае часто оказывается, что результат сильно отличается от ис-

ходного замысла. Поэтому мы считаем, что уже в начальной школе необходимо познакомить ребят с планированием, как важной частью полноценной проектной деятельности (проект 7), которая проходит в несколько этапов. Большинство авторов описывают схожие этапы проектной деятельности, хотя называют и описывают их несколько по-разному [9, 7, 5, 6]. Учитывая специфику проектной деятельности в среде Scratch, мы выделим следующие её этапы.

Подготовительный этап. На этом этапе происходит постановка цели (конечного результата деятельности); составляется план деятельности: выделяются все объекты предстоящего проекта, их свойства и взаимодействия; выделяются отдельные подзадачи и последовательность их выполнения.

Организационный этап — распределение ролей в группе по виду деятельности (художник, программист, музыкальный редактор и т. п.) или по подзадачам.

Осуществление проекта. На этом этапе разрабатывается визуальное представление объектов и их скрипты. Здесь же происходит отладка кода.

Презентация проекта и рефлексия — демонстрация проекта классу, обсуждение и оценивание проекта; формулирование выводов.

При оценивании итогового проекта следует обращать внимание на такие элементы проекта, как:

1. наличие заставки и титров с указанием авторства;
2. наличие соответствующего музыкального сопровождения с указанием в титрах авторов музыки;
3. продуманность интерфейса игры;
4. наличие этапа подведения итогов игры;
5. художественное оформление;
6. техническую сложность;
7. защиту от ошибок;
8. практическую значимость проекта.

Помимо собственно проекта следует оценивать умения групповой работы. В выборе критериев такого оценивания мы ориентируемся на опыт работы Центра образования

«Школа здоровья» (Москва) [3]. Умение организовывать работу в группе следует оценивать по:

1. наличию и функциональности разделения обязанностей;
2. информированности группы о результатах работы;
3. вкладу каждого члена группы.

Представленные рекомендации к проведению занятий содержат следующие разъяснения:

- материал, введённый при разработке проекта (новые понятия среды и языка программирования Scratch, из математики и информатики);
- комментарии для учителя;
- скрипты базового и расширенного проектов (примерные).

Проект 1. Знакомимся со Scratch. Наш Кот ходит и мяукает!

Цель: знакомство со средой и языком программирования Scratch.

Тип проекта: анимация.

Продолжительность: 3 часа.

Направления развития проекта

1. Бесконечное движение Кота с отражением от стен и сменой костюмов
2. Кот мяукает при ударе о стену;
3. при нажатии на стрелку вверх Кот меняет цвет или применяется другой эффект;
4. при нажатии на стрелку вниз эффект должен отмениться;
5. при нажатии на пробел Кот говорит «Привет!»;
6. при щелчке мышью по Коту он мурлычет.

Введённый материал и пояснения

Scratch

Элементы интерфейса: сцена; группы блоков; стиль вращения; вкладки скриптов, костюмов и звуков; кнопки Старт и Стоп; смена направления движения спрайта;

изменение имени спрайта; режимы просмотра (малый экран, полный экран, режим демонстрации); главное меню (пункт FILE).

Типы блоков: обычные (stack blocks); С-образные (C-shaped «mouth»); шапочки (hats); логические датчики.

Группы блоков:

контроль: `всегда` (бесконечный цикл); `когда щёлкнут по зелёному флажку; когда клавиша [ ] нажата; ждать [ ] секунд; всегда, если < >; когда щёлкнут по { }`.

движение: `идти [ ] шагов; если край, оттолкнуться`.

внешность: `следующий костюм; изменить [ ] эффект на [ ]; говорить [ ] в течение [ ] секунд; думать [ ] [ ] секунд`.

звук: `играть звук [ ]; играть звук [ ] до завершения`.

сенсоры: `касается [ ]`.

Математика

Десятичная дробь. Отрицательное число.

Информатика и ИКТ

Понятия исполнителя, системы команд исполнителя (СКИ), алгоритма, линейного алгоритма, цикла, события, параллельности

Поля ввода: обычное и со списком.

Меню: стандартное горизонтальное, контекстное меню. Меню блоков.

Обратите внимание

Спрайт можно перетаскивать мышкой по сцене.

Выполнить скрипт (или команду) можно двойным щелчком по нему (по ней). Причём команду можно выполнить прямо в панели команд, не перетаскивая её в панель скриптов.

Скрипт, выполняющийся в данный момент, имеет белое обрамление.

Текущее направление движения спрайта показано синим отрезком около изображения спрайта в информаци-

онной панели вверху экрана. Его можно изменить при помощи перетаскивания мышкой конца отрезка.

Хорошим тоном является изменение имени спрайта по умолчанию на имя, подходящее к Вашему проекту. Наш спрайт можно назвать, например, «Кот»: русские буквы поддерживаются. В результате изменятся и названия некоторых блоков, и содержимое выпадающих списков. Скажем, блок `когда щёлкнут по {Sprite 1}` станет называться `когда щёлкнут по {Кот}`.

Без команды `если край, оттолкнуться` спрайт убежит за край экрана.

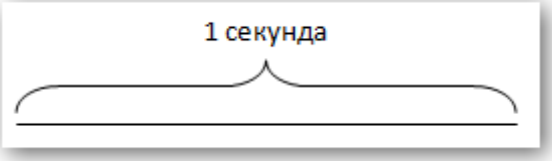
С этой командой спрайт может отражаться тремя способами: можно повернуть (при этом спрайт перевернётся вверх ногами), только поворот влево-вправо, не поворачивать. Это зависит от выбранной кнопки на панели текущего спрайта (слева от спрайта).

Если щёлкать по костюмам спрайта, его изображение на сцене меняется соответственно.

Команда `ждать [ ] секунд` требует понимания понятия десятичной дроби. Это понятие довольно легко ввести, если сначала использовать целые числа для задержки. Убедившись вместе детьми, что с задержкой в 1 секунду (наименьшее целое число) Кот движется слишком отрывисто, подводим их к понятию десятичного числа.

«Вы видите, что целая секунда — это очень много для нас…»

Нарисовать на доске прямую линию — секунду.



«Как нам подставить в поле ввода число меньшее одной секунды? Давайте разделим эту секунду на 10 равных частей (или долей, как дольки мандарина)…»

Разделить линию-секунду на десять частей.

*«И возьмём **ноль** целых секунд...»*

*Записать на доске **0**.*

*«Поставим запятую, чтобы показать, что теперь мы будем брать **десятые доли** этой секунды…»*

Поставить запятую после нуля.

«А теперь запишем столько десятых долей секунды, сколько нам нужно, например, 3…»

*В итоге получится запись **0,3**.*

*«Это число читается так: **ноль целых три десятых**. То есть наша задержка будет составлять ноль целых секунд и три десятых доли секунды».*

При работе с командой `изменить [ ] эффект на [ ]` дети сталкиваются с отрицательными числами. Здесь можно ограничиться представлением отрицательного числа как противоположного положительному: сначала изменили эффект (например, цвет) на 30, а потом вернём обратное значение, т. е. изменим эффект на –30. Позднее, при работе над проектами, требующими использования понятия системы координат, к отрицательным числам следует вернуться на другом уровне (см. проект 11).

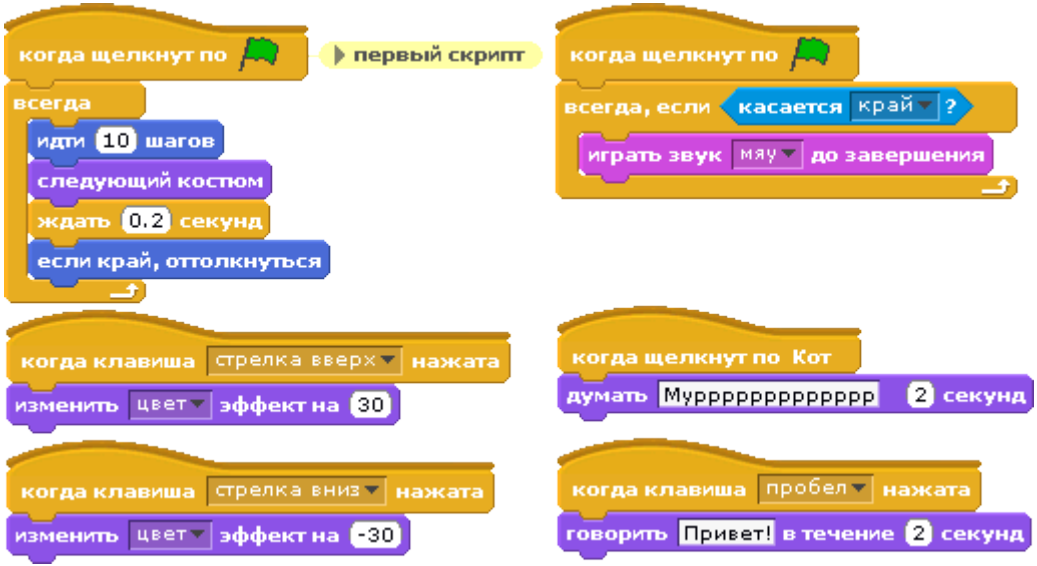
Одним из наиболее существенных моментов при создании первого проекта является введение понятия параллельности. Из курса математики первого класса

дети уже знакомы с понятием параллельных прямых. К сожалению, сложно провести удачные аналогии между параллельностью в геометрии и программировании. Удобнее всего опираться на понятия об одновременности и последовательности действий.

Следует противопоставить параллельное и последовательное выполнение действий примерами из окружающей жизни: мы можем одновременно идти и говорить (параллельное выполнение), нам не нужно прерывать одно из действий, чтобы выполнить второе. Но мы не можем одновременно отреза́ть кусок хлеба и намазывать на него масло (это требует последовательного выполнения действий).

В Scratch команды одного скрипта реализуют последовательное выполнение (следование). Команды разных скриптов одного объекта реализуют параллельное выполнение команд этим объектом. Здесь особое внимание обратим на следующий момент. Продемонстрировать последовательное выполнение команд было бы естественно через добавление команды `играть звук [мяу]` в скрипт движения Кота. Однако внешне это будет выглядеть как параллельное выполнение и у ребят может возникнуть путаница с пониманием последовательного и параллельного выполнения команд. Это объясняется особенностями команд `играть звук [ ]` и `играть звук [ ] до завершения`. Мы настоятельно рекомендуем учителю до урока (без детей) поэкспериментировать с этими командами и «почувствовать» разницу между ними.

На уроке ребятам нужно сразу предложить составить отдельный скрипт мяуканья с проверкой условия касания края. При этом сначала используйте команду `играть звук [ ]`, убедитесь вместе с детьми, что Кот мяукает «неправильно». Следует разобрать (или объяснить), отчего это может происходить: из-за задержки в первом скрипте Кот успевает проверить касание края несколько раз, и соответственно несколько раз начинает произносить звук. После этого замените команду на `играть звук [ ] до завершения` и отметьте разницу.



Листинг 1. Пример реализации бесконечного движения Кота
с отражением от стен и дополнительными эффектами

Если убрать команду ожидания из скрипта движения, то перестанет корректно обрабатываться проверка условия касания края в скрипте мяуканья.

В конце каждого занятия обязательно с о х р а н я й т е п р о е к т ы через меню FILE / СОХРАНИТЬ ПРОЕКТ. По умолчанию проекты сохраняются в папку SCRATCH PROJECTS (местоположение папки — МОИ ДОКУМЕНТЫ). Желательно сразу приучить детей пользоваться кнопкой СОЗДАТЬ НОВУЮ ПАПКУ в окне сохранения программы.

ПРОЕКТ 2. СКА́ЧКИ. ЩЕКОЧЕМ ЛОШАДКУ

Цель: закрепление изученного с дополнительным продвижением в изучении интерфейса среды.

Тип проекта: интерактивная анимация.

Продолжительность: 3 часа.

Направления развития проекта

1. Лошадка скачет вперёд-назад;
2. добавить звук «галопа»;
3. при ударе Лошадки о край раздаётся ржание;
4. при щелчке мышью по Лошадке она говорит «Ой, щекотно!»;
5. изменить последний скрипт так, чтобы Лошадка говорила «Ой, щекотно!» при касании указателя мыши.

Введённый материал и пояснения

Scratch

Элементы интерфейса: панель инструментов (кнопки удаления, роста, сжатия спрайта)  ; кнопка выбора нового спрайта; кнопка импорта костюма для спрайта; кнопка импорта звуков; меню FILE / ОПИСАНИЕ ПРОЕКТА…

Информатика

Ввод и редактирование текста при описании проекта.

Обратите внимание

Любой звук после выбора можно прослушать (кнопка с треугольником).

Имена костюмов можно менять, так же как и имена спрайтов (вкладка «костюмы»).

Полезно для каждого проекта создавать о п и с а н и е , в которое включается имя и класс автора, дата последнего изменения (или номер версии), правила управления.

Листинг 2. Пример реализации проекта 2 (вариант 1 соответствует п. 4, вариант 2 — п. 5)

Ещё раз обратить внимание детей на параллельность выполнения некоторых действий. Цикл основного скрипта движения и цикл проигрывания звука галопа *всегда* выполняются одновременно. Это особенно хорошо видно во время

работы программы: все скрипты, выполняющиеся в данный момент, имеют белое обрамление.

Проект 3. Играем на пианино и других музыкальных инструментах

Цель: знакомство с музыкальными возможностями Scratch.

Тип проекта: музыкальный.

Продолжительность: 2 часа.

Направления развития проекта

1. Используя блоки, `Когда клавиша [ ] нажата` и `ноту [ ] играть [ ] тактов`, создать аналог пианино.
2. Добавить возможность выбора инструмента.
3. Поиграть на своём инструменте.
4. Изменить программу так, чтобы «инструмент» играл аккордами.

Введённый материал и пояснения

Scratch

Элементы интерфейса: новых нет.

Группы блоков:

звук: `ноту [ ] играть [ ] тактов; выбрать инструмент [ ].`

Основы музыкальной грамоты

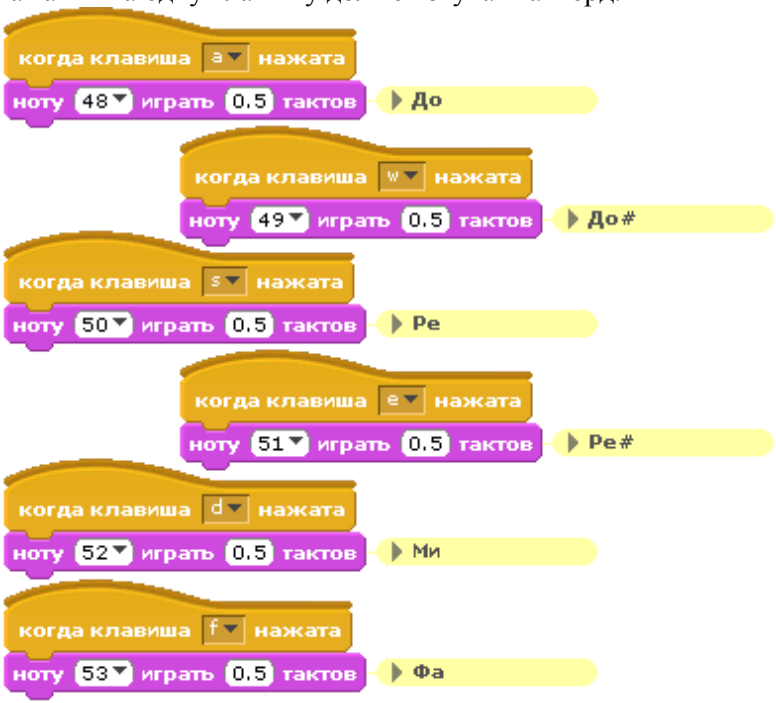
Нота, такт, длительность ноты, тон, диез, бемоль, бекар.

Обратите внимание

Это очень простой проект, который вызывает массу положительных эмоций у детей (листинг 3). Если дети не знакомы с основами музыкальной грамоты, у них могут возникнуть вопросы о том, что за число стоит в поле «такт». Не углубляясь в музыкальную терминологию, достаточно сказать, что это длительность ноты. Для имитации работы пианино это число менять не нужно, оно должно быть одним и тем же для всех нот. В следующем проекте мы вернёмся к этому понятию.

Более полная имитация пианино получится, если использовать и чёрные клавиши, для обозначения которых вводятся понятия диеза (#) и бемоля (♭).

Обязательно следует выделить время для игры на «инструменте». Интересно, что можно играть аккордами: аккордный звук извлекается одновременным нажатием на несколько клавиш. Можно сделать имитацию аккордеона: при нажатии на одну клавишу должен звучать аккорд.



Листинг 3. Фрагмент программы имитации пианино (проект 3)

Проект 4. Записываем и сочиняем музыку

Цели: (1) научиться записывать музыку с нот; (2) развитие творчества.

Тип проекта: музыкальный.

Продолжительность: 3 часа.

Ход работы и направления развития проекта

1. Создать новый проект «Песенка».
2. Составить программу проигрывания мелодии «Чижик-пыжик» (рис. 1).
3. Добавить ударные инструменты.
4. Создать «мини-оркестр», играющий припев известной детской песенки «Мы едем, едем, едем» (рис. 12).
5. При желании сочинить свою мелодию.

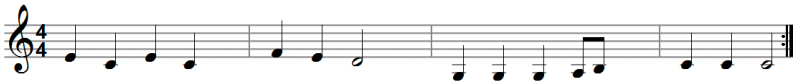


Рис. 1. Песенка «Чижик-пыжик»

Введённый материал и пояснения

Scratch

Группы блоков:

контроль: `повторить [ ];`

звук: `установить громкость [ ]; барабану [ ] играть [ ] тактов; оставшиеся [ ] тактов.`

Основы музыкальной грамоты

Нота, такт, размер такта, длительность ноты, реприза, пауза.

Обратите внимание

Если для имитации пианино не требовалось менять длительность нот, то для записи даже самой простой мелодии не обойтись без понятия такта, размера такта, длительности ноты.

Заметим, что в Scratch один такт соответствует целой длительности. Поэтому команда, например, `ноту [60] играть [0,5] тактов` означает проигрывание ноты *До* половинной длительности.

Команда `оставшиеся [ ] тактов` означает паузу заданной длительности.

Чаще всего музыкальные размеры тактов обозначаются по количеству четвертей, содержащихся в одном такте. В таблице 3 приложения 1 (стр. 102) приведены обозначения

нот и пауз различных длительностей и их количество в одном такте наиболее простых размерностей: $\frac{2}{4}$, $\frac{3}{4}$ и $\frac{4}{4}$.

Сначала необходимо научить ребят записывать в Scratch мелодии с уже готовых нот. Для этого надо следует подготовить раздаточный материал: таблицу обозначения длительности нот и пауз, рисунок соответствия названия нот и их положения на нотном стане (рис. 9, прил. 1) и обязательно нотные записи различных мелодий.

После того, как дети справятся с песенкой «Чижик-пыжик» (лист. 4) можно переходить к составлению программы проигрывания произвольной мелодии. Для этого надо научиться записывать мелодию в виде нот. Здесь могут помочь записи нот с их названиями и следующие пояснения.

Каждый музыкальный такт начинается с так называемой сильной доли. Например, в известной песенке «Маленькой ёлочке/ Холодно зимой» сильные и слабые доли будут распределяться по слогам и тактам так: «**МА**-лень-кой» (такт); «**Ё**-лоч-ке» (такт); «**ХО**-лод-но зи-» (такт); «-**МОЙ**» пауза (такт) и т. д. Музыкальный акцент делается на сильную долю. Тогда нотная запись этой мелодии будет выглядеть как на рис. 2 (предложите детям записать её самостоятельно).

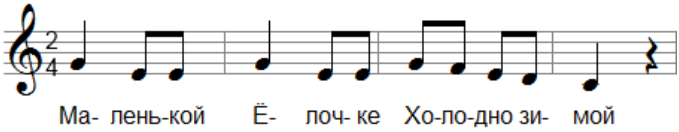


Рис. 2. Нотная запись мелодии «Маленькой ёлочке»

Следует очень чётко отсчитывать такты, иначе мелодия может не получиться.

До сих пор в Scratch для повторения выполнения каких-либо действий использовался бесконечный цикл. Если мелодия содержит повторяющиеся фрагменты можно воспользоваться циклом, повторяющим действия конечное число раз (в отличие от бесконечного цикла `всегда`). В нотной записи повторение обозначается знаком репризы  .

Листинг 4. Мелодия «Чижик-пыжик»

Листинг 5. Мелодия «Маленькой ёлочке»

Громкость инструментов различна, поэтому её можно и нужно регулировать.

Один объект может проигрывать мелодии только одним инструментом. Например, можно составить программу, в которой Кот играет все партии песенки «Тра-та-та» (рис. 12) одним инструментом. Но будет лучше, если каждую партию этой мелодии будет играть отдельный инструмент. Для каждого инструмента понадобится отдельный объект. Обратите внимание, команда `ноту [ ] играть [ ] тактов` ограничена двумя октавами. Но в некоторых случаях (как, например, для второй партии мелодии «Тра-та-та») могут понадобиться очень низкие звуки. Их можно получить простым подсчётом полутонов от одной известной ноты до другой. Например, мы хотим выяснить номер ноты СОЛЬ (на рис. 3 соответствующая клавиша окрашена голубым цветом), зная,

что нота ДО (соответствующая клавиша окрашена жёлтым цветом) имеет номер 60. Между этими нотами 5 полутонов (промежутков между клавишами). Тогда, $60 - 5 = 55$. Это и есть тот номер, который нужно вписать в команду как в обычное поле ввода.

Рис. 3. Интерфейс команды проигрывания нот в Scratch

Добавив ударные инструменты, получим целый оркестр. Однако слишком большая численность оркестра, а также большое число повторений приводит к рассогласованию игры «музыкантов». Это связано с тем, что параллельность процессов в компьютерных системах достигается за счёт квантования процессорного времени: фактически мы имеем дело с псевдо-параллельностью. К вопросу согласования игры музыкантов мы вернёмся чуть позже, после введения понятия сообщения.

Проект 5. Ска́чки-2

Цель: изучение взаимодействия объектов на основе обмена сообщениями.

Тип проекта: анимация с обработкой событий.

Продолжительность: 4 часа.

Ход работы и направления развития проекта

Проект основан на проекте Ска́чки.

1. Добавить новый объект *lion1-a*.
2. Изменить имя объекта на «Лев».
3. Добавить Льву костюм *lion1-b*.
4. При ударе Лошадки о край Лев должен сказать «Ах ты, бедняжка!».
5. Лошадка отвечает Льву, поддерживая диалог.

Партия флейты

Партия гитары

Партия ударных

Листинг 6. Пример имитации трио «Гитара–флейта–ударные» (проект 4)

Введённый материал и пояснения

Scratch

Элементы интерфейса: новых нет.

Группы блоков:

контроль: `передать [ ]; когда я получу [ ]`.

Информатика

Понятие сообщения, обработчика сообщения.

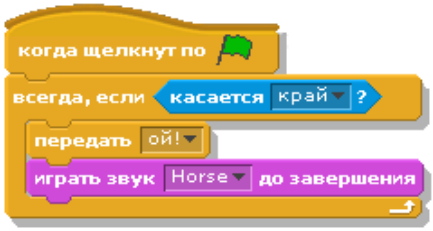
Обратите внимание

Технология обмена сообщениями — одна из наиболее важных в объектно-ориентированном программировании.

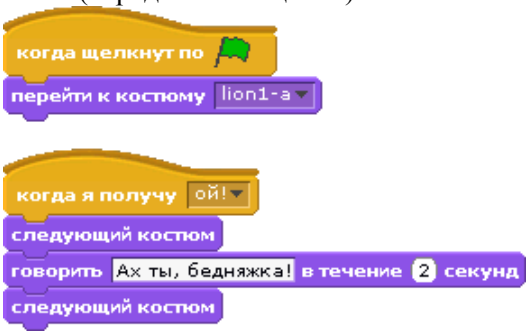
При задании имени сообщения можно использовать русские буквы.

Название сообщения должно иметь смысл.

В блоке `когда я получу [ ]` можно выбрать только уже существующие сообщения.



Изменённый скрипт Лошадки (передача сообщения)



Скрипты Льва

Листинг 7. Пример реализации передачи и обработки сообщений (проект 5)

Проект 6. Используем слои

Цели: (1) закрепление изученного материала; (2) научиться перемещать объекты в различные слои.

Тип проекта: анимация.

Продолжительность: 2 часа.

Ход работы и направления развития проекта

1. Открыть проект Ска́чки-2.
2. Добавить новый объект *rock* (гора) из папки *Things* (вещи). Переименовать объект в *Гора*. Разместить гору в центре сцены и увеличить размер горы при помощи кнопки роста спрайтов.
3. Используя команды `перейти в верхний слой` и `перейти назад на [ ] слоёв` добиться, чтобы Лошадка пробегала позади горы, когда щёлкнут по горе, или перед горой, когда щёлкнут по сцене.

Введённый материал и пояснения

Scratch

Элементы интерфейса: новых нет.

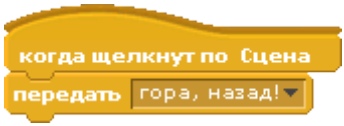
Группы блоков:

внешность: `показаться; спрятаться; перейти в верхний слой; перейти назад на [ ] слоёв`.

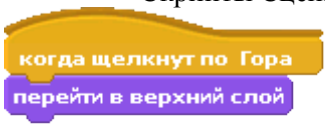
Обратите внимание

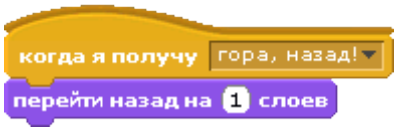
Перед выполнением проекта следует показать детям возможность перемещения объекта в верхний слой можно прямо на сцене, щёлкнув по нему мышкой и слегка потянув. Можно добавить большее количество новых объектов и показать, что в таком случае размещение объектов на нужном уровне при помощи мышки довольно затруднительно.

Объект можно спрятать и заставить его появиться в нужный момент.



Скрипты Сцены





Скрипты горы

Листинг 8. Работа со слоями (проект 6)

Проект 7. Планируем и делаем мультфильмы и комиксы (свободное проектирование)

Цели: (1) знакомство с этапами проектирования; (2) развитие творчества.

Тип проекта: анимация, динамический комикс.

Продолжительность: 6 часов.

Введённый материал и пояснения

Проектная деятельность

Этапы проектной деятельности: постановка цели, составление плана, реализация, проверка (тестирование программы, анализ результатов).

Обратите внимание

Используя полученные знания, ребята могут создавать свои собственные содержательные мультфильмы динамические комиксы, интерактивные открытки.

Необходимо приучить детей к составлению чёткого плана предстоящего проекта на бумаге. План должен содержать:

1. цель в виде рассказа или рисунка с пояснениями;

2. выделение всех имеющихся объектов, их свойств (костюмов и звуков) и поведения (действий и взаимодействий).

При выполнении проекта может возникнуть необходимость в новых блоках, ещё не изученных возможностях Scratch. Нам кажется, что правильно будет поддержать авторов, показав им несколько новых приёмов для реализации собственных задумок, выходящих за рамки.

Проект 8. Создаём свой объект в графическом редакторе

Цель: научиться создавать собственные спрайты и анимировать их.

Тип проекта: анимация.

Продолжительность: 4 часа.

Ход работы и направления развития проекта

1. Нарисовать новый объект: нераспустившийся цветок (бутон).
2. Добавить этому объекту несколько (не менее трёх) костюмов, изображающих стадии раскрытия цветка: от бутона (костюм № 1) до полностью раскрытого цветка (последний костюм).
3. Произвести анимацию, последовательно сменяя костюмы объекта с небольшой задержкой.
4. Добавить звуковое сопровождение.
5. Добавить восход солнца. Распускание цветочка происходит только утром.

Введённый материал и пояснения

Scratch

Элементы интерфейса: кнопка РИСОВАТЬ НОВЫЙ ОБЪЕКТ  ; информационная панель объекта.

Графический редактор в Scratch: выбор цвета в палитре; инструменты «кисточка», «ластик», «заливка», «прямоугольник», «эллипс», «линия», «пипетка»; выбор размера кисти; кнопки ОТМЕНА (undo, отменить по-

следнее действие), ОТМЕНА (redo, отменить отмену), ОЧИСТИТЬ; установка центра вращения объекта.

Информатика и ИКТ

Инструменты графического редактора, приёмы создания изображения, способы создания анимации. Циклы.

Обратите внимание

Создание собственных анимированных объектов — это сильный мотив для ребят, ориентированных на художественное творчество. Для них программистский аспект будет достаточно сложен. Не следует форсировать их развитие в области программирования.

Инструменты редактора можно рассматривать безотносительно к проектированию.

Для создания изображений удобно пользоваться графическим планшетом.

Объект «Солнце» имеет один костюм. Для реализации восхода Солнца не строго по вертикали, а по диагонали, ему следует сменить направление движения — схватить мышью синий вектор направления на информационной панели и повернуть его против часовой стрелки (рис. 4). По умолчанию объект имеет направление 90 (движение вправо).

Если дети ещё не знакомы с понятием системы координат, следует возвращать Солнце в исходную позицию (например, в левый нижний угол сцены) путём простого перетаскивания мышкой.

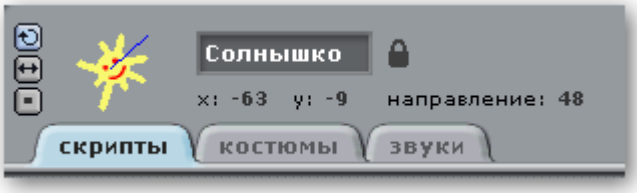


Рис. 4. Информационная панель объекта «Солнышко»

Первый костюм цветка придётся сделать «с нуля». Но создание каждого следующего костюма нужно проводить на основе предыдущего. Для этого на вкладке «Костюмы» надо нажать кнопку КОПИРОВАТЬ у самого нижнего костюма.

Появится копия этого костюма, у которой теперь можно нажать кнопку РЕДАКТИРОВАТЬ (рис. 5). Удалив при помощи ластика некоторые детали изображения и дорисовав необходимые, получим новый костюм.

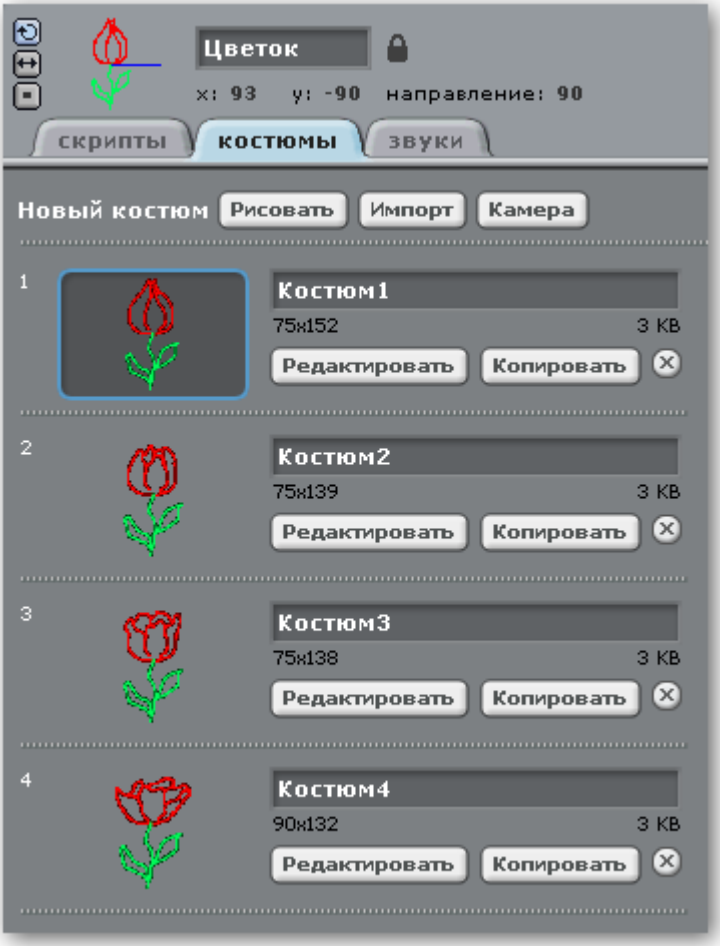
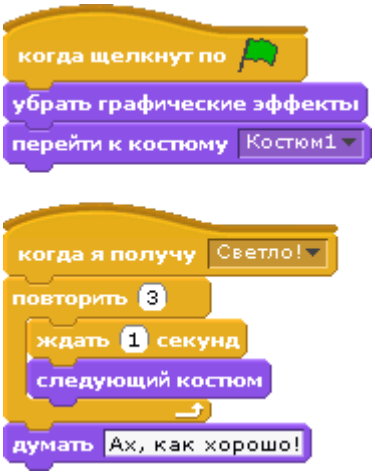
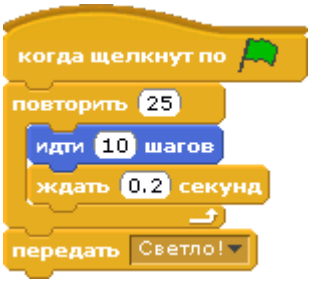


Рис. 5. Вкладка костюмов нового объекта



Скрипты Цветка



Скрипт Солнышка

Листинг 9. Пример реализации проекта 8

Проект 9. Анимируем полёт пчелы

Цель: научиться создавать костюмы к готовым объектам папки Costumes.

Тип проекта: анимация.

Продолжительность: 2 часа.

Ход работы и направления развития проекта

1. Создать новый проект.
2. Добавить новый спрайт *Animals\bee1*.
3. Изменить имя спрайта на «Пчела».
4. Перейти на вкладку Костюмы.
5. Создать копию костюма (кнопка «копировать»).
6. Нажать кнопку «редактировать» нижнего костюма.
7. В окне редактора воспользоваться кнопкой «выбор» для выделения крыльев Пчелы.
8. Воспользоваться кнопкой «поворот против часовой стрелки».
9. Установить крылья на место.

10. В случае необходимости воспользоваться ластиком или кнопкой «отмена».
11. Аналогично создать третий костюм.
12. Известным способом создать анимированный полёт.
13. Сохранить проект под именем «Пчёлка».

Введённый материал и пояснения
Scratch

Элементы интерфейса: новых нет.

Графический редактор в Scratch: кнопка «выбор», «поворот против часовой стрелки», «поворот по часовой стрелке», «Установить центр костюма».

Обратите внимание

Если центры костюмов не совпадают, то при анимации возможны неестественные скачки. Центр вращения устанавливается в графическом редакторе под палитрой.

Даже если всё сделано правильно, движение Пчелы остаётся «рваным» из-за того, что взмахи крыльями (смена костюмов) и перемещение выполняются последовательно. Разрешить проблему плавности движения постараемся в одном из следующих проектов.

Проект 10. Создаём оркестр (синхронизируем многоголосье)

Цели: (1) закрепить навыки работы с графическим редактором; (2) закрепить навыки синхронизации скриптов при помощи сообщений; (3) закрепить навыки создания музыкальных композиций.

Тип проекта: музыкальный.

Продолжительность: 4 часа.

Направления развития проекта

1. Составить программу, в которой проигрывается многоголосная (оркестровая) музыкальная композиция. Использовать не менее трёх–четырёх инструментов.
2. Добавить анимацию, изображающую дирижёра и музыкантов.
3. Попробовать создать один проект силами нескольких команд, каждая из которых отвечает за кодирование

только своей партии. Объединить все партии в одном проекте.

Введённый материал и пояснения

Нового материала нет.

Обратите внимание

Чтобы составить одноголосную мелодию на Scratch достаточно иметь самое общее представление о записи музыки. При наличии некоторого чувства ритма с этим может справиться практически любой школьник. Задача существенно усложняется при попытке создать многоголосие. Можно использовать нотную запись любой мелодии, ребята могут приносить ноты своих любимых произведений. Здесь следует обратить внимание детей на следующее: если в нотной записи встречается аккорд, то исполнение его каждого звука относится к отдельной партии. Поэтому такие мелодии следует заранее разложить по партиям. Несколько партий одного инструмента может исполнять один объект, но партии различных инструментов должны принадлежать разным объектам.

Если дети не занимаются в музыкальной школе, то можно использовать несложный вариант песенки «Тра-та-та» (прил. 1).

Сначала следует программу проигрывания мелодии основной партии (рис. 11). Затем добавляются вторая и третья партия (рис. 12), а также (при желании) ударные. При этом возникает проблема рассинхронизации нескольких инструментов: даже при создании не очень длинных (15–20 секунд) некоторые партии уйдут вперёд, некоторые отстанут. Это связано с «псевдомногозадачностью» операционной системы, которая на самом деле выполняет программы и скрипты не параллельно, а последовательно чередуя их небольшие фрагменты. Как уже отмечалось выше (проект 4), необходимо синхронизировать скрипты, чтобы избежать их рассогласования. Для этого введём в программу «дирижёра», который будет указывать всем участникам оркестра, когда надо начинать играть очередной фрагмент партии.

Практически это выглядит следующим образом. Дирижёр рассылает сообщения при помощи «передать и ждать».

Каждое сообщение инициирует выполнение (проигрывание) одного или двух тактов произведения. За такое небольшое время рассогласование партий ещё не успеет произойти. Поэтому каждый фрагмент будет восприниматься как следует. Однако, возможны небольшие, на грани восприятия, задержки между тактами, вызванные как раз необходимостью синхронизации.

Пример такой организации можно посмотреть в приложении 2.

Проект 11. Создаём плавные анимации

Цели: (1) познакомиться с понятием «система координат» и научиться соотносить движение спрайта с системой координат Scratch; (2) закрепить понятие параллельности потоков.

Тип проекта: анимация.

Продолжительность: 6 часов.

Ход работы

1. Открыть проект «Пчёлка».
2. Сохранить проект под новым именем.
3. Удалить команды «идти 10 шагов», «если край, оттолкнуться».
4. Создать новый скрипт, в котором осуществляется движение пчелы в случайную точку экрана через команду «плыть».

Введённый материал и пояснения

Scratch

Элементы интерфейса: датчики координат мышки, датчики координат текущего спрайта, главное меню (пункт File / Сохранить как).

Группы блоков:

движение: `идти в x: [ ] y: [ ]; плыть [ ] секунд в точку x: [ ] y: [ ]; установить x в [ ]; установить y в [ ].`

операторы: `выдать случайное от [ ] до [ ].`

Математика

Понятия отрицательного числа, системы координат (СК), случайного числа.

Обратите внимание

Вначале следует вернуться к понятию отрицательного числа на примере обычного термометра. Здесь важно, чтобы школьник понимал, что отрицательные числа противоположны положительным: то есть сложение положительного числа с отрицательным равносильно вычитанию из положительного числа модуля отрицательного числа. Затем можно сказать, что термометр — это пример координатной оси. А если взять два термометра — вертикальный и горизонтальный и расположить их так, чтобы нули у них совпадали, то получим пример системы координат.

Полезно давать детям задания типа: «Установить мышку (или спрайт в виде крестика) в координаты (35, –48)». Удобно загрузить фон «xy-grid» изображающий систему координат с координатной сеткой (папка Backgrounds).

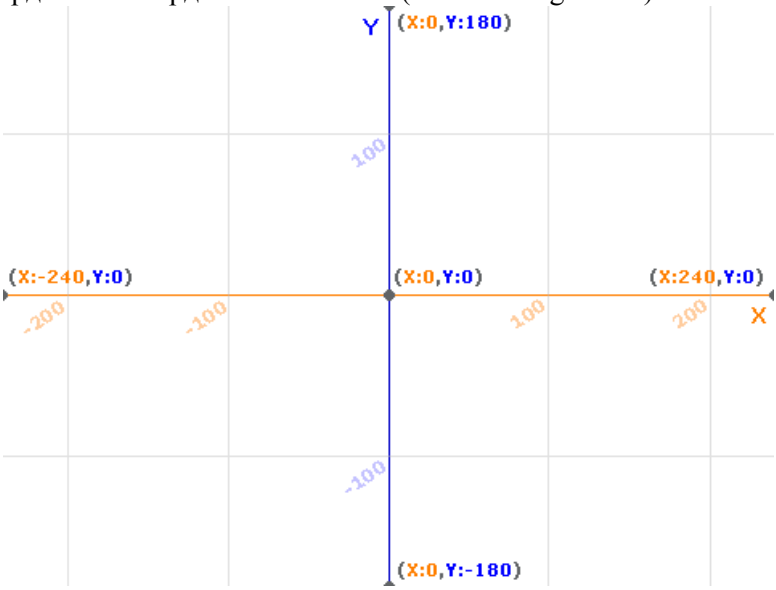


Рис. 6. Фон «xy-grid» системы координат с координатной сеткой (папка Backgrounds)

Можно также поиграть в морской бой пакета Роботландия, шашки, шахматы. Имеются удобные педагогические программные средства (например, «Страна Фантазия» С. Н. Тур и Т. В. Бокучава) для закрепления навыков работы с системой координат. Отметим, что досконально изучение системы координат не является основной целью данных занятий. Достаточно изучить эту тему на том уровне, который позволяет выполнять проекты.

Следует обратить внимание детей на то, что изменение координат связано с движением объекта.

В результате создания первого скрипта Пчела, стоя на месте, будет махать крыльями. Далее объясняем детям, что теперь нужно, чтобы Пчела не просто махала крыльями, но ещё и плавно перемещалась в случайную точку на экране. Как правило, дети сами придумывают такой вариант решения: в уже имеющуюся конструкцию добавить дополнительный блок скольжения.

Запуск программы показывает, что Пчела действительно летает случайным образом, однако взмахи крыльями у неё происходят только в момент смены направления, а не в процессе полёта. Тогда мы предлагаем завести ещё один бесконечный цикл «всегда» и вынести в него скольжение Пчелы в случайную точку. И поскольку оба бесконечных цикла — взмахи крыльями и случайные скольжения — должны выполняться одновременно (параллельно), обе конструкции должны приводиться в действие по нажатию на зелёный флажок. В результате один объект «Пчела» будет выполнять одновременно два потока (листинг 10).

Получили плавное движение Пчелы. Дальнейшее решение задачи предусматривает поворот Пчелы в направление движения путём сравнения её старых и новых *x*_координат. Эта возможность будет рассмотрена позже, после введения понятия переменной (проект 15).

С методической точки зрения лучше оставить прежнюю Пчелу и сделать точную её копию, для которой и выполнить указанные действия. Тогда можно понаблюдать за поведением и старого, и нового объектов и сравнить их визуально.

Листинг 10. Плавное движение Пчелы в случайную точку экрана
с использованием параллельности потоков

При введении понятия случайного числа можно привести в качестве примера бросание кубика, монеты. Также можно попробовать обсудить правила нанесения ударов в игре *«Advanced Dungeons & Dragons»*. Этот последний пример может послужить толчком к созданию множества новых проектов.

Проект 12. Изменяем Кота в зависимости от окружающих условий

Цель: знакомство с командами ветвления.
Тип проекта: анимация с элементами ИИ.
Продолжительность: 3 часа.

Ход работы направления развития проекта

1. Создать (нарисовать) новый фон сцены с красными и жёлтыми кругами разной величины.
2. Уменьшить размер Кота при помощи кнопки Сжать спрайт панели инструментов.
3. Создать скрипт движения Кота с отражением от стен. Если во время движения Кот касается красного цвета, должен измениться какой-либо эффект Кота (например, мозаика).
4. Изменить скрипт таким образом, чтобы в то время, пока Кот не касается красного цвета, изменялся другой эффект (или тот же эффект изменялся на противоположное значение).
5. Добавить в проект новый объект — Бабочку, перемещающуюся случайным образом. Если Кот касается красного круга, изменить какой-либо эффект, если касается жёлтого круга, изменить другой эффект, если ничего не касается, Кот должен повернуться к Бабочке.

Введённый материал и пояснения
Scratch

Группы блоков:

движение: `если [ ]; если [ ] или [ ]; повернуться к [ ].`

сенсоры: `касается цвета [ ]`.

Информатика

Команды ветвления.

Обратите внимание

Правильное понимание команды ветвления очень важно для дальнейшей работы в Scratch. Ветвление — это всем (так или иначе связанным с программированием) известное англоязычное `if-then-else`. В русской традиции полное наименование этой команды — `если-то-иначе`. В Scratch слово «то» («then») опущено. Кроме того, перевод этой команды на русский язык несколько неудачен: вместо `иначе` используется `или`, что несколько затрудняет понимание. По смыслу это именно «иначе», что есть противопоставление «если»: если условие выполняется, то должна реализоваться одна серия команд, а если *не* выполняется (иначе) — другая. Следует уделить этому моменту особое внимание.

Вначале (п. 3) потребуется сокращённый вариант команды: `если [касается цвета (красный)]`. Цвет выбирается пипеткой:

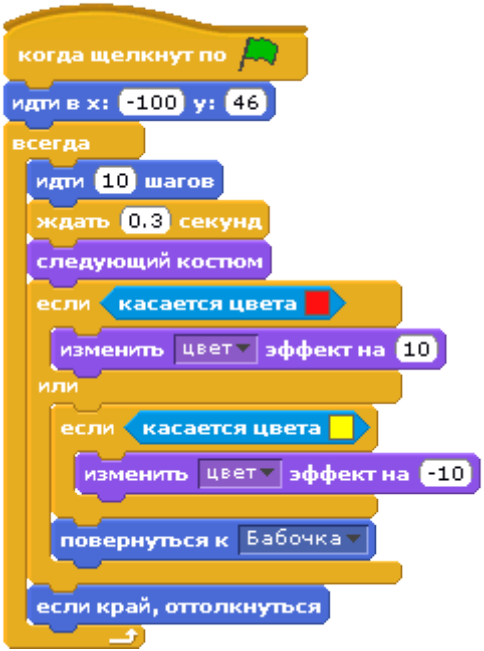
- щёлкнуть мышкой в окошке цвета команды `касается цвета`;
- щёлкнуть мышкой в том месте экрана, который содержит нужный цвет (в нашем случае — по красному кругу на сцене).

После изучения действий Кота можно изменить условие выполнения команды или саму команду в теле `если [ ]` и понаблюдать за изменениями.

Далее можно поставить следующую задачу: если Кот касается красного круга, то эффект мозаики (или любой другой) должен увеличиться, а если не касается, то эффект мозаики должен уменьшиться. Для этого потребуется команда `если [ ] или [ ]`.

Более сложный вариант задачи (п. 5) потребует использования вложенных команд ветвления. Следует переименовать новый объект в «Бабочка». В команде `повернуться к [ ]` выбрать из списка «Бабочка».

Листинг 11. Пример использования команд ветвления (проект 12)



Листинг 12. Пример организации вложенного ветвления (проект 12, п. 5)

Проект 13. Создаём мультфильмы и комиксы (свободное проектирование)

Цели: (1) закрепление изученного материала; (2) закрепление этапов планирования; (3) включение в деятельность обсуждения проектов; (4) развитие навыков самоконтроля.

Тип проекта: анимация, комикс.

Продолжительность: 12 часов.

Обратите внимание

Особое внимание следует уделить использованию системы координат и случайных чисел.

Желательно, чтобы разрабатываемые проекты были не слишком велики и завершались за два–три занятия (за неделю).

С этого момента можно организовывать общий просмотр и обсуждение проектов.

Проект 14. Знакомимся с переменными

Цели: (1) познакомиться с задачами, в которых возникает необходимость в переменных; (2) познакомиться с группой блоков *переменные*.

Тип проекта: анимация с элементами ИИ.

Продолжительность: 6 часов.

Ход работы и направления развития проекта

1. Предварительное задание: разработать программу, в которой Кот ходит по экрану и отражается от его границ.
2. Предложить детям сделать так, чтобы Кот отразился только пять раз, после чего он должен остановиться и сказать: «Ох, устал…».
3. Для этого придётся завести переменную — счётчик касаний, которая при каждом отражении от стены увеличивается на единицу.

4. Удобно использовать блок `если < > или < >` для различения состояний «касания» и «некасания» (листинг 13).
5. Добавьте в проект мяч, скачущий по полю и отражающийся от стен.
6. Посчитайте, сколько раз во время своего движения Кот коснётся Мяча.

Введённый материал и пояснения

Scratch

Элементы интерфейса: бегунок (слайдер), флажок (переключатель, чекбокс).

Группы блоков:

контроль: `если < > или < >; остановить скрипт, остановить всё; ждать до < >.`

переменные: кнопка «создать переменную», `поставить [ ] в [ ]; изменить [ ] на [ ]; показать переменную [ ]; спрятать переменную [ ].`

операторы: `[ ]=[ ]; не < >.`

Обратите внимание

Понятие переменной — одна из наиболее важных концепций языков программирования.

Переменные нужны для того, чтобы хранить значения, изменяющиеся при выполнении программы.

Переменные имеют *имя* и *тип*. В Scratch очень слабая типизация, но по содержимому или по способу использования переменной (по контексту) можно догадываться, к какому типу она принадлежит. Это может быть число, строка или список.

Переменные могут задаваться или только для одного спрайта (*локальные*; доступ к ним осуществляется только из скриптов данного спрайта), или для всех (*глобальные*; их можно использовать из любых скриптов). Всегда следует подумать, какой тип переменной выбрать. В основном возникает необходимость в использовании *локальных* переменных: это так называемые *поля* объектов. Глобальные переменные нужны реже и работа с ними требует особенной ак-

куратности.

Переменные можно отображать на экране, а можно скрыть отображение. Следует приучить детей включать отображение переменных, чтобы в дальнейшем легче осуществлять отладку программы.

Для проверки количества касаний стены понадобится оператор сравнения [ ]=[ ]. Операторы сравнения знакомы детям с курса математики и не вызывают трудностей. Отметим, что можно поменять оператор внутри блока, используя правую кнопку мыши (рис. 7).

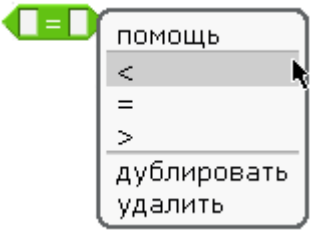


Рис. 7. Контекстное меню оператора сравнения

Добавить мяч в проект просто (не забудьте изменить мячу направление движения), но правильно подсчитать касания — уже сложнее. Здесь нельзя использовать конструкцию вида «если Кот касается Мяча, то прибавляй к счётчику единицу». Поскольку касание двух спрайтов может длиться довольно долго, всё это время значение счётчика будет увеличиваться (этот вариант обязательно нужно реализовать вместе с детьми). При касании спрайтов нужно увеличивать счётчик касаний, после чего организовать цикл вида

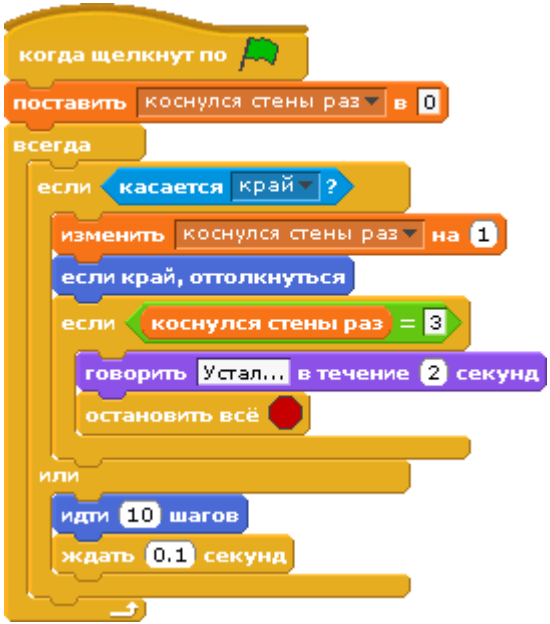
.

Проект 15. Разворачиваем Пчелу в направление движения (развитие проекта 11)

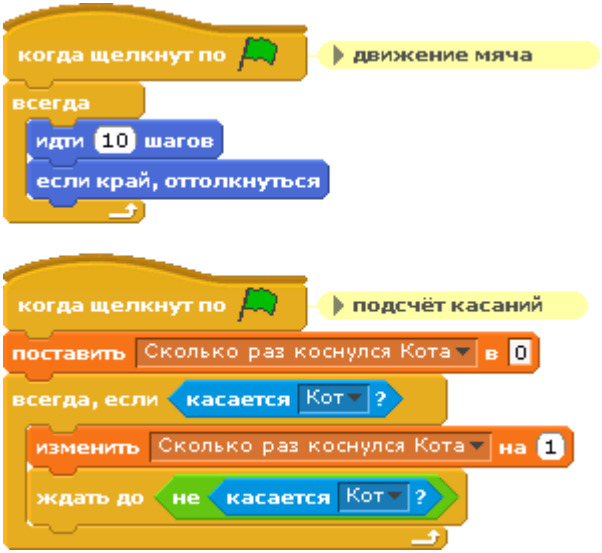
Цели: (1) закрепить понятие переменной; (2) закрепить понятие системы координат.

Тип проекта: анимация.

Продолжительность: 2 часа.



Скрипт Кота
(подсчёт количества касаний стены)



Скрипты Мяча
(подсчёт количества касаний Кота)

Листинг 13. Подсчёт количества касаний Кота и Мяча (проект 14)

Ход работы

1. Нарисовать новый объект Точка, используя цвет фона.
2. Создать две глобальные переменные *x* и *y*, принимающие случайные значения.
3. Изменить программу так, чтобы Пчела поворачивалась к Точке и перемещалась в новые координаты.

Введённый материал и пояснения

Scratch

Группы блоков:

движение: `повернуть в направление [90]; положение x.`

операторы: `[ ]>[ ]`.

Обратите внимание

Очевидно, что поворачивать Пчелу следует, сравнивая её старую и новую *x*-координаты. Несмотря на то, что программа получится совсем короткая (листинг 15), детям этот вариант покажется довольно сложным для понимания. Поэтому для поворота Пчелы можно прибегнуть к «хитрости»: создать дополнительный объект — точку, окрашенную в цвет фона (для того, чтобы её не было видно). Точка помещается в новые координаты *(x,y)*. Пчела поворачивается к этой точке (листинг 14). Обратите внимание на использование команды `передать [ ] и ждать`.

Более старшим или развитым детям (особенно тем, кто хочет в дальнейшем серьёзно заниматься программированием) вариант с двумя координатами (старыми и новыми) следует объяснить и попытаться его реализовать. Для этого нужно будет воспользоваться командой `повернуть в направление [90]`. Несмотря на то, что в Scratch до сих пор не использовалось понятие поворота, эта команда довольно легко воспринимается детьми, так как в выпадающем списке рядом с углом поворота подписано направление (рис. 8).

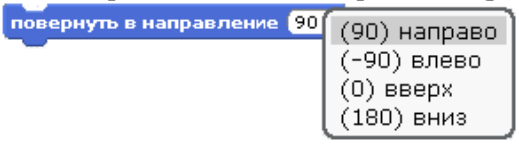
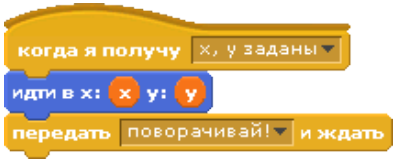


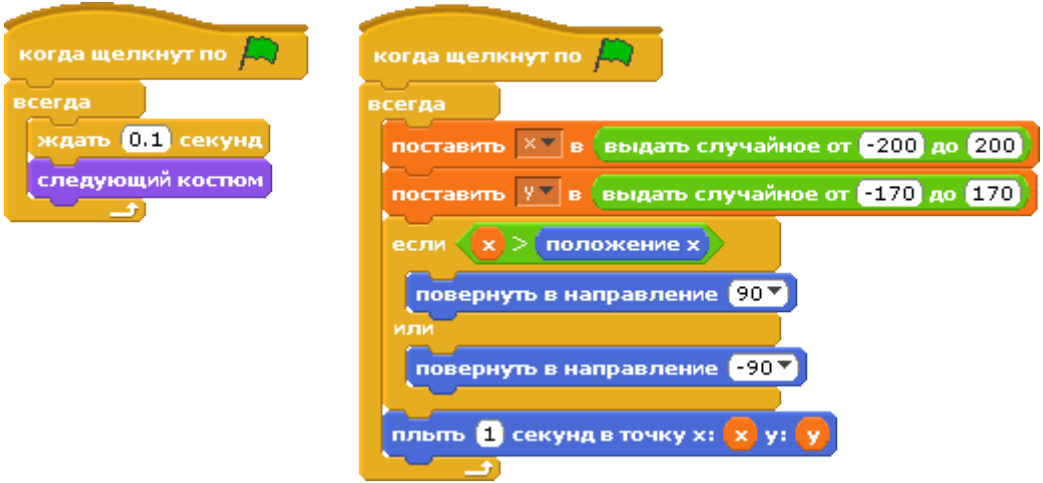
Рис. 8. Команда движения `повернуть в направление [90]`

Скрипты Пчелы



Скрипт Точки

Листинг 14. Пример поворота Пчелы в направлении движения



Листинг 15. Пример поворота Пчелы в направление движения путём сравнения старых и новых *x*-координат

Проект 16. Делаем мультфильмы, комиксы, игры (свободное проектирование)

Цели: (1) развитие творчества; (2) закрепление планирования в виде составления таблицы объектов, их свойств и взаимодействий; (3) развитие умений коллективной работы (распределение ролей, задач, навыков взаимодействия); (4) развитие чувства ответственности; (5) постепенный переход к более сложным проектам.

Тип проекта: анимация, динамический комикс, игра.

Продолжительность: 12 часов.

Примерные темы проектов

1. Анекдоты.
2. Интерактивные сюжетные комиксы «Человек-паук», «Идём на дискотеку», «Шрек» и т. д.
3. Мультфильмы «Красная Шапочка», «Подводные гонки», «Супер-Зайчик спасает мир», «Магическая битва» и т. д.
4. Анимация с элементами случайности и взаимодействием объектов. Сюжет может быть, например, следующим. В случайном месте экрана появляется сыр. Мышь со случайной скоростью бежит к сыру, а Сова со случайной скоростью летит к Мышке. Если Мышь успела добежать до сыра раньше, чем Сова схватила её, то Сова разворачивается и улетает. В противном случае Сова издаёт победное уханье (идея Алёны Пюра, г. Оренбург, гимназия № 3, адрес проекта http://scratch.mit.edu/projects/Apyura/614815).
5. Интерактивные игры: «Собери бананы», «Проведи Мышку по лабиринту», «Автогонки».

Введённый материал и пояснения

Scratch

Элементы интерфейса: меню File / Импорт проекта...

Обратите внимание

Если сюжет проекта увлекает ребёнка, то проектирование приобретает характер познавательной деятельности,

поскольку ребёнок обязательно сталкивается с необходимостью использования ещё не изученных блоков и понятий. Чем увлекательнее сюжет, тем более глубоко ребёнок продвинется в познавательном процессе.

При работе над сложными проектами возможна работа в малых группах. В таком случае необходимо помочь детям распределить роли в группе и разбить задачу на подзадачи. Каждую подзадачу может выполнять отдельный ребёнок в отдельном проекте. Впоследствии проекты можно объединить путём импортирования (меню FILE / ИМПОРТ ПРОЕКТА…), поэтому следует вместе с детьми заранее хорошо продумать взаимосвязи между объектами. Если для детей этот момент окажется сложным, то лучше будет вернуться к такому способу групповой работы позже (см. проект 26) и предложить ребятам поработать за одним компьютером.

ПРОЕКТ 17. ИЗУЧАЕМ ПОВОРОТЫ

Цели: (1) познакомиться с градусной мерой углов; (2) познакомиться с группой блоков *перо* (аналог языка Logo).

Тип проекта: графика.

Продолжительность: 3 часа.

Ход работы и направления развития проекта

1. Изменить спрайт.
2. Добавить в панель скриптов из группы *движение* блоки `идти [ ] шагов` и `повернуть в направление [ ].`
3. Изменить величину шага на 50.
4. Убедиться, что двойной щелчок по блоку приводит к выполнению соответствующей команды.
5. Поменять направление поворота спрайта и выполнить команду двойным щелчком.
6. Чередуя повороты в разных направлениях с движением прямо провести Кота по периметру экрана.
7. Опустить перо при помощи соответствующей команды группы *перо*.

8. Выполнить задание, аналогичное (6): нарисовать лесенку из правого верхнего в левый нижний угол экрана.

Введённый материал и пояснения

Scratch

Группы блоков:

движение: `повернуться на [ ] градусов`, `повернуть в направление [ ]`.

перо: `очистить`, `поднять перо`, `опустить перо`.

Математика

Поворот. Направление поворота. Угол. Градусная мера угла. Прямой угол. Полный оборот (360°).

Обратите внимание

Вначале учитель должен рассказать про повороты, углы и градусы, а затем продемонстрировать эти понятия на доске и в среде Scratch.

Задание реализуется в режиме непосредственного выполнения команд, а не в режиме программирования.

Следует сменить спрайт на *cat2*, *crab1* или *crab2*, которые больше подходят для выполнения предлагаемых заданий.

Система угловых координат в Scratch задаётся следующим образом:

0° — вверх;

90° — вправо (направление по умолчанию);

180° — вниз;

–90° — влево.

Повернуться в каком-либо их этих направлений можно при помощи блока `повернуть в направление [ ]`. Использование этого блока приводит к повороту в *абсолютном направлении*.

Существует и *относительный поворот*, отсчитываемый от текущего направления спрайта. Поворот спрайта может осуществляться как влево, так и вправо; для каждого направления поворота имеется свой блок. В то же время можно использовать отрицательные углы поворота: в этом случае поворот, например, вправо на (–20°) тождественен

повороту влево на (+20°).

Проект 18. Создаём своего исполнителя

Цели: (1) закрепить понятия градусной меры угла и поворота; (2) вспомнить понятие исполнителя.

Тип проекта: графика.

Продолжительность: 3 часа.

Задание

Создать исполнителя «Кот-художник» со следующей системой команд:

1. идти 10 шагов;
2. поднять и опустить хвост (перо);
3. повернуться влево и вправо;
4. изменить цвет пера;
5. изменить тень (оттенок) пера;
6. изменить размер пера.

Ход работы и направления развития проекта

1. Изменить спрайт на *cat2*.
2. Назвать спрайт «Художник».
3. После шапки «Когда щёлкнут по зелёному флажку» записать условия инициализации (установка по центру, очистка экрана, сброс цвета и размера пера и т. п.)
4. Добавить блок `когда клавиша [ ] нажата` и выбрать в его выпадающем списке клавишу «1» (сначала нужно будет выбрать пункт «больше», а потом уже единицу).
5. Пристыковать к шапке `когда клавиша [1] нажата` блок `идти [10] шагов`. Теперь при нажатии на клавишу «1» на клавиатуре Кот-художник будет делать десять шагов вперёд.
6. Аналогично нужно добавить скрипты для остальных функций исполнителя. Мы предлагаем реализовать, как минимум, следующие:
   - стрелка влево: поворот на 10^{o} влево;
   - стрелка вправо: поворот на 10^{o} вправо;
   - стрелка вниз: опустить перо;

- стрелка вверх: поднять перо;
- клавиша «s»: изменить размер пера на 1;
- клавиша «a»: изменить тень пера на 10;
- клавиша «пробел»: изменить цвет пера на 10.

7. Добавить дополнительные клавиши управления с новыми функциями.
8. Сделать симметричное управление для двух объектов на экране с одной клавиатуры. Можно устроить соревнования между собой или «в четыре руки» между парами за разными компьютерами.
9. Более сложный проект: управление несколькими объектами при помощи одних и тех же клавиш. Здесь нужно будет организовать переключение между объектами (например, при помощи клавиши TAB). То есть лишь один из объектов будет активным, а остальные — пассивными.

Введённый материал и пояснения
Scratch

Группы блоков:

контроль: `когда клавиша [ ] нажата.`

Обратите внимание

Понятие «исполнитель» впервые было использовано в самом первом проекте. Затем оно использовалось, явно или неявно, при работе со всеми другими проектами. Здесь можно ещё раз вспомнить понятие «исполнитель» и «система команд исполнителя». Выполняя предлагаемый проект, ученик узнает, как обрабатывать в программе нажатия на различные клавиши, и в результате получит эффективное средство для ручного управления экранными объектами.

С точки зрения программирования проект очень прост. Однако с точки зрения управления он может быть сложен (это зависит от количества добавленных функций). Нужно поуправлять исполнителем вручную, нарисовав несколько картинок разной сложности. Полезно провести аналогии между ручным и программным управлением, сопоставить набор ручных операций, необходимых для создания изображения, с соответствующей программой.

Проект будет частично работоспособен даже без запуска на выполнение (без нажатия на зелёный флажок), потому что нажатие клавиш вызывает срабатывание соответствующего скрипта-обработчика.

Scratch умеет обрабатывать только ограниченный набор клавиш и совсем не может обрабатывать комбинации клавиш.

Одна из наиболее интересных идей здесь — реализация работы на одном компьютере вдвоём (и более) и управление несколькими объектами. Проекты могут быть ориентированы на сотрудничество или соперничество.

Проект 19. Изменяем направление движения в зависимости от условия

Цели: (1) закрепить понятие градусной меры угла; (2) вспомнить команды ветвления.

Тип проекта: графика с элементами ИИ.

Продолжительность: 2 часа.

Проект основан на проекте 12.

Ход работы

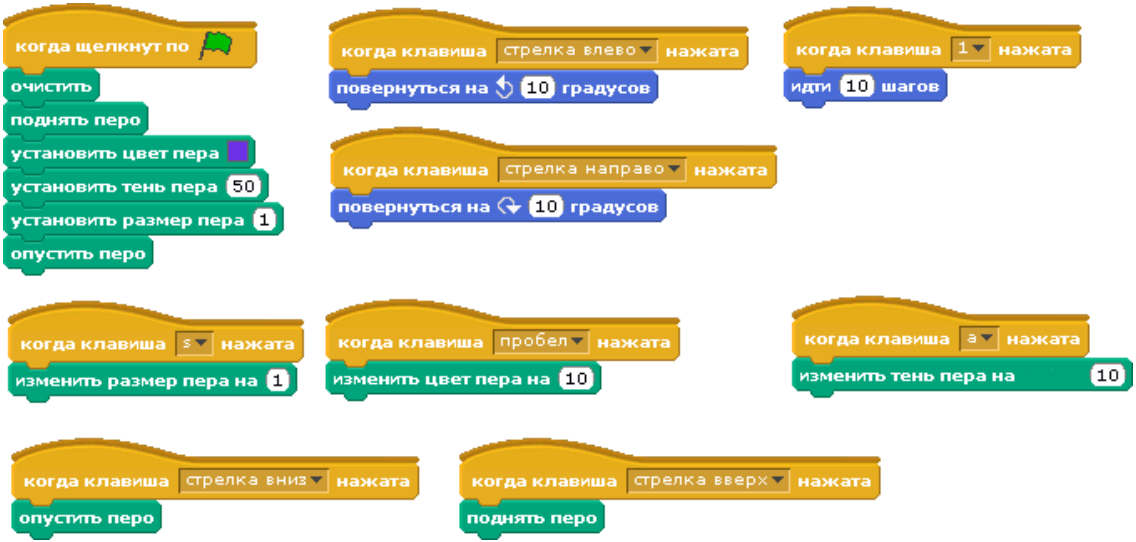
1. Сохранить проект под новым именем.
2. Изменить скрипт следующим образом: если Кот касается красного круга, выполнить поворот против часовой стрелки на 15 градусов, иначе, если Кот касается жёлтого круга, выполнить поворот по часовой стрелке на 15 градусов.
3. Добавить команды `очистить` и `опустить перо`.
4. Открыть проект 12.

Введённый материал и пояснения

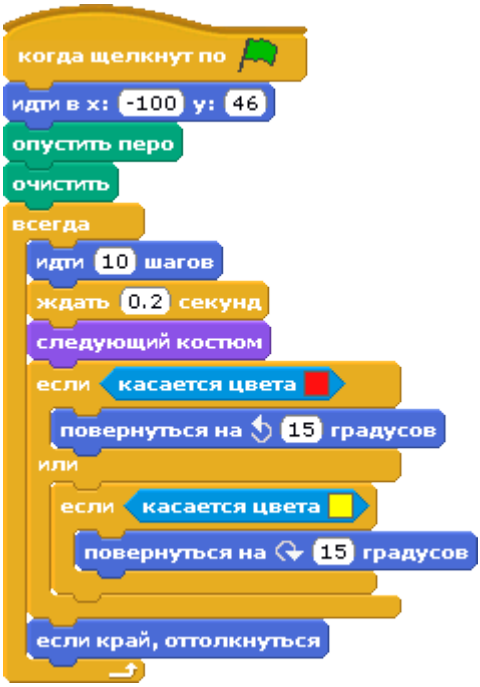
Нового материала нет.

Обратите внимание

Выполнение проекта не вызывает сложностей. Добавление команд из группы *перо* позволит получить интересные эффекты. Можно предложить детям организовать конкурс на самый интересный след Кота, используя повороты на разные углы и разные условия.



Листинг 16. Пример создания исполнителя «Кот-художник» (проект 18)



Листинг 17. Пример реализации изменения направления движения в зависимости от условий

Проект 20. Рисуем разноцветные геометрические фигуры

Цели: (1) закрепить понятие градусной меры угла; (2) изучить средства рисования группы *перо*; (3) познакомиться с выражением единиц в процентах; (4) познакомиться с правильными геометрическими фигурами и изучить способы их рисования.

Тип проекта: графика.

Продолжительность: 4 часа.

Направления развития проекта

1. Составить программу, в которой Кот перемещается в случайное место экрана и рисует квадрат со сторонами разного цвета и разной толщины.

2. Изменить программу так, чтобы Кот рисовал квадраты, правильные треугольники, шестиугольники, окружности, снежинки.
3. В зависимости от толщины линии должен меняться размер Кота.
4. Создать несколько объектов (в нашем примере будет рассмотрен случай с четырьмя котами), рисующих в случайных местах экрана разные геометрические фигуры по очереди. То есть каждый следующий Кот должен начать рисование своей фигуры только после того, как закончил рисовать предыдущий.

Введённый материал и пояснения

Scratch

Группы блоков:

перо: `установить цвет пера [ ], изменить цвет пера на [ ], установить размер пера [ ].`

сенсоры: `мышка нажата?`

внешность: `установить размер на [ ] %.`

Математика

Проценты. Правильные многоугольники.

Обратите внимание

В этом проекте впервые появляется понятие «процент». По всей видимости, лучше всего не вдаваться в математические подробности, а опереться на понятие дроби. Тем не менее, на это придётся отвести время.

Имеется разница между командами «изменить» и «установить».

Использование случайных чисел приводит к тому, что поведение программы не повторяется от запуска к запуску. Необходимо задать начальное направление Кота (например, 90^0), и уже тогда подбирать диапазоны случайных координат Кота. Если этого не сделать, фигуры могут «ломаться».

Следует обратить внимание ребят на скорость перемещения Кота и соотнести её с преодолеваемым расстоянием.

Собственно рисование квадрата лучше оформить в виде отдельного скрипта — процедуры, которая вызывается

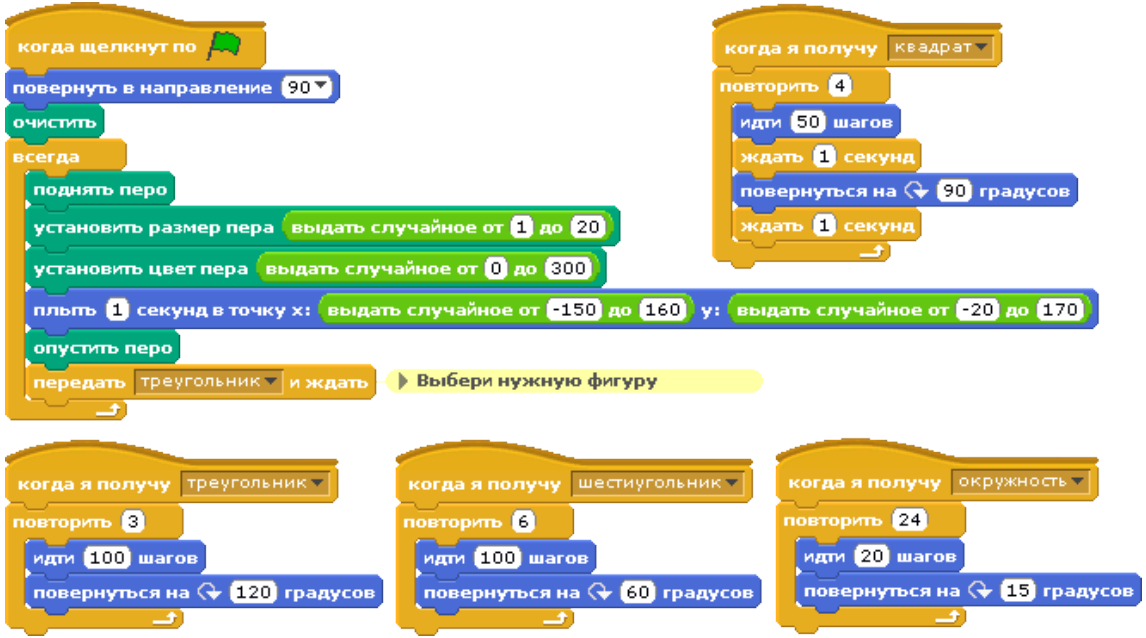
по соответствующему сообщению. Это позволит в дальнейшем легко создавать процедуры для рисования других фигур путём дублирования и изменения числа повторов и угла поворота.

С методической точки зрения в процедуру рисования каждой конкретной фигуры желательно поставить задержку (в нашем примере задержка добавлена в процедуру квадрата), чтобы отчётливо был виден каждый шаг Кота. Это позволит детям лучше понять геометрию фигур.

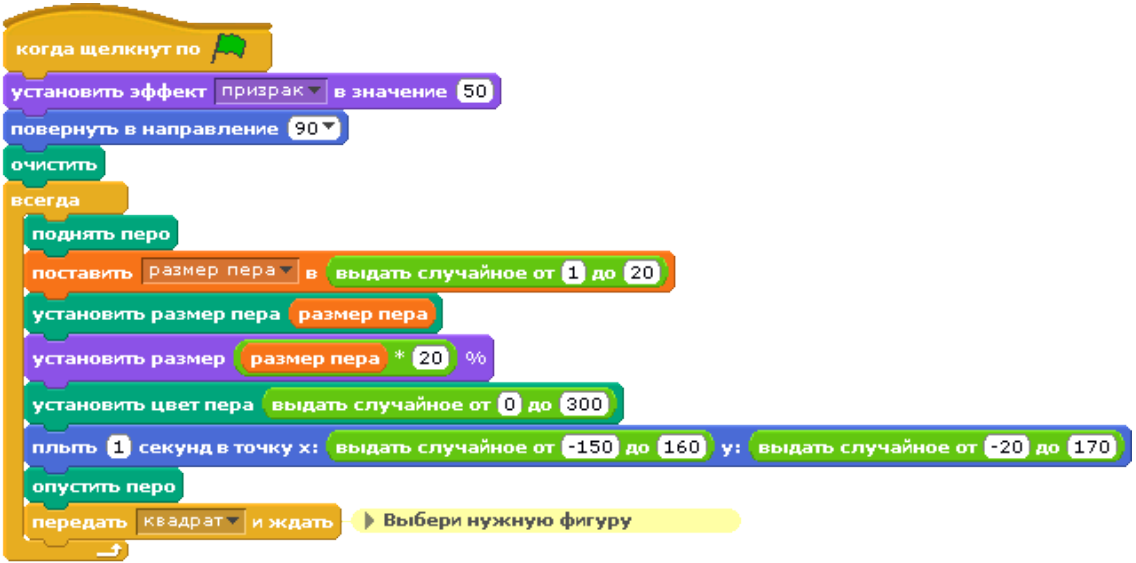
Изменение размера Кота в зависимости от толщины линии требует введения переменной, поскольку генерация случайного числа из определённого диапазона каждый раз происходит заново (иными словами, для задания размера Кота нужно использовать уже установленный случайный размер пера, а не генерировать случайное число ещё раз). Важно при этом, что переменная должна быть локальной.

Поскольку размер Кота может быть достаточно большим, полезно воспользоваться командой `установить эффект [призрак] в значение [50]`, чтобы Кот стал прозрачным.

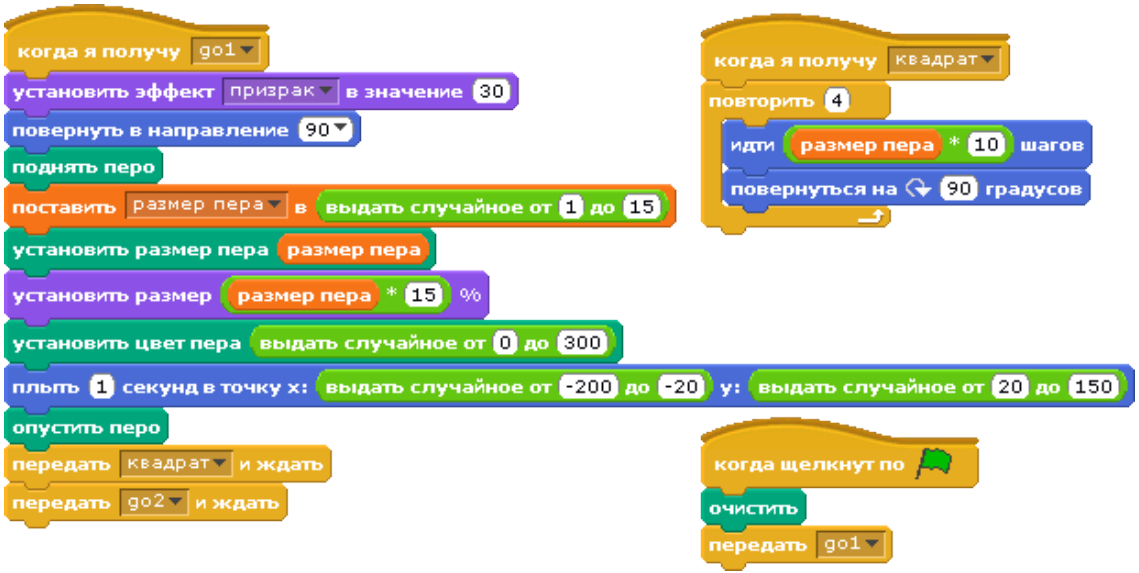
Для реализации рисования по очереди следует воспользоваться передачей сообщений. Сначала следует убрать команду цикла (повторение будет реализовано другим способом). Новые объекты создаются дублированием (через контекстное меню). Оставить каждому объекту по одной процедуре, после чего можно запустить программу на выполнение. В результате несколько котов разного размера одновременно будут рисовать различные геометрические фигуры в разных местах сцены. Далее необходимо у всех объектов удалить команды `очистить` и `когда щёлкнут по`  . Организовать рисование «по эстафете» через передачу сообщений: Кот1 рисует, например, квадрат, шлёт сообщение Коту2; Кот2 обрабатывает сообщение, рисует свою фигуру и передаёт сообщение Коту3 и так далее. Последний объект передаёт сообщение первому объекту. Тогда, для того, чтобы первый объект вообще начал работу, после нажатия зелёного флажка он должен всё очистить и передать сообщение самому себе.



Листинг 18. Кот рисует разные геометрические фигуры различного цвета и толщины в случайном месте экрана



Листинг 19. Изменение размера Кота в зависимости от толщины линии (проект 20**,** п. 3)



Листинг 20. Скрипты объекта Кот1 (проект 20, п. 4)

Желательно ставить задачу не до конца, а дать только самое общее описание. Это позволит ребятам додумать детали и разработать вполне оригинальные проекты.

Проект 21. Рисуем натюрморт, пейзаж, портрет (свободное проектирование)

Используя блоки группы *перо* нарисовать любое изображение: предмет, натюрморт, пейзаж, портрет.

Цели: (1) развитие творчества; (2) закрепление этапов планирования.

Тип проекта: графика.

Продолжительность: 12 часов.

Обратите внимание

Поскольку мы занимаемся не просто художественным творчеством, оцениваться должен не только конечный результат (рисунок). Некоторые ребята могут выполнить проект так, что само возникновение рисунка будет интересно. Это следует отметить: динамические проекты интереснее.

Проект 22. Создаём самую настоящую игру

Идея проекта: Герой собирает предметы, разбросанные в случайных местах игрового поля. За это ему начисляются очки. Если собраны все предметы, то звучит победная музыка.

Цели: (1) закрепление понятия переменной; (2) изучение планирования в виде составления таблицы объектов, их свойств и взаимодействий.

Тип проекта: игра.

Продолжительность: 12 часов.

Направления развития проекта.

Добавить переход с уровня на уровень.

Введённый материал и пояснения

Обратите внимание

Это плохо поставленная задача, несмотря на то, что многие дети играли в подобные игры. Очень важно не забыть, что мы не просто пишем программу, но разрабатываем проект. То есть нужно выполнить все этапы проектирования, про-

тестировать проект, сделать отчёт и презентацию, провести защиту.

Проект 23. Кот анализирует сложную окружающую обстановку

Цель: изучить логические операции и соответствующие им блоки в разделе *операторы*.

Тип проекта: с элементами ИИ.

Продолжительность: 2 часа.

Ход работы и направления развития проекта

1. Используйте фон из проекта 12.
2. Написать программу, в которой Кот ходит по сцене. Если во время движения Кот коснулся только красного круга, то он должен произнести «Какое всё красное!»
3. Если Кот коснулся только жёлтого круга, то он должен произнести «Какое всё жёлтое!».
4. Если Кот коснулся одновременно жёлтого и красного кругов, то он должен произнести «Какое всё цветное!»
5. Вместо Кота добавить объект Круг, имеющий костюмы следующих расцветок: красный, жёлтый, зелёный, красно-жёлтый, красно-зелёный, жёлто-зелёный и красно-жёлто-зелёный. Круг должен менять свой костюм в соответствии с теми цветами, которых он касается.

Введённый материал и пояснения

Scratch

Элементы интерфейса: новых нет.

Группы блоков:

операторы: `< > и < >; не < >`.

Информатика

Логические операции.

Обратите внимание

В этом проекте мы сознательно не используем логическую операцию «или», поскольку сложно придумать содержательный пример её использования. Это вызвано тем, что

в быту мы, произнося *«или» (OR)*, в большинстве случаев имеем в виду *исключающее «или» (XOR)*. Например, говоря, что мы поедем в кино на троллейбусе или на автобусе, мы имеем в виду что воспользуемся каким-то одним видом транспорта. При этом использование операции *«или»* предполагает, что имеется возможность воспользоваться двумя видами транспорта одновременно. Заметим, что операция *исключающее «или»* в Scratch отсутствует, но можно обойти это ограничение, используя имеющиеся функции.

Листинг 21. Пример реализации проекта 23

Проект 24. Создаём игры (свободное проектирование)

Цели: (1) развитие творчества; (2) закрепление планирования в виде составления таблицы объектов, их свойств и взаимодействий; (3) развитие умений коллективной работы (распределение ролей, задач, навыков взаимодействия) и чувства ответственности.

Тип проекта: игра.

Продолжительность: 18 часов.

Возможна (но необязательна) работа в малых группах.

Введённый материал и пояснения

Нового материала нет.

Обратите внимание

Любой проект, но игровой — в особенности, требует к себе внимания аудитории. Автор должен поделиться достижением и побывать в состоянии успеха и одобрения. Поэтому нужно отвести время (после завершения презентаций) на игру с чужими проектами. Здесь, вероятно, будет не только эмоциональное, но и интеллектуальное общение. Ребята обычно выясняют детали того или иного проекта, берут на вооружение приёмы, выработанные товарищами. Эти «беседы в кулуарах конференции», как считает большинство учёных, могут быть даже более полезны, чем сама конференция в силу неформальности общения и его диалогичности.

Проект 25. Организуем диалог с пользователем

Цели: (1) изучить тип данных «строка»; (2) познакомиться с группой строковых блоков в разделах *операторы* и *сенсоры;* (3) научиться использовать строки при создании диалоговых проектов.

Тип проекта: интерактивный.

Продолжительность: 10 часов.

Ход работы и направления развития проекта

Написать простую программу, ведущую диалог с пользователем от имени Кота.

1. Определить две строковые переменные: «Собеседник» и «Строка».
2. Кот должен поздороваться и спросить имя у пользователя. Это имя нужно поместить в переменную «Собеседник».
3. При помощи блока `слить [ ] [ ]` составить различные фразы, включающие имя собеседника.
4. При помощи других блоков узнать, чьё имя длиннее.
5. Проверить, не начинаются ли имя собеседника и имя Кота на одну и ту же букву.

Введённый материал и пояснения
Scratch

Группы блоков:

операторы: `слить [ ] [ ]; буква [ ] в [ ]; длина строки [ ]; [ ]или[ ]`.

сенсоры: `спросить [ ] и ждать, ответ`.

Информатика

Логические операторы.

Обратите внимание

Дети уже знакомы с логическими операциями *«и»* и *«не»*. Новой оказывается логическая операция *«или»*. Сложности использования этой операции обсуждались в проекте 23. В настоящем проекте операция *«или»* также используется в значении *исключающего «или»*, но здесь это не может привести к ошибке, поскольку нельзя в строку ввести сразу два варианта ответа: «отлично» и «хорошо».

В Scratch (начиная с версии 1.4) можно вставлять имена переменных в большинство блоков, где это допускается по смыслу. Например, возможен такой вариант: 

. В этом случае Кот скажет: «Истина», поскольку истинно приведённое равенство для строк.

Если использовать строковые значения в математических выражениях вместо чисел, то возможно два варианта реакции программы: (1) если строка изображает число по всем правилам, то над ней будут выполнены числовые действия; (2) если строка — это просто последовательность символов, то её значение принимается равным нулю.

Если использовать числовые выражения вместо строк, то они рассматриваются как строки. Например, аналог вывода на печать числовой переменной *x* может быть таким: `говорить [x] в течение [5] секунд`.

Можно легко обращаться к цифрам числа как к элементам линейного массива при помощи блока `буква [ ] в [ ]`.

В качестве закрепления навыков работы со строковыми переменными можно разработать два проекта:

1. Вычисление суммы цифр введённого пользователем числа. Усложнённый вариант — вычисление арифметического корня числа (если полученная сумма цифр числа также является числом не однозначным, то вычисления должны быть продолжены).
2. Программа — угадыватель задуманного пользователем числа (из заранее оговорённого диапазона; например, [1…100]).

Листинг 22. Пример реализации проекта 25

Проект 26. Создаём игры и творческие проекты (свободное проектирование).

Цели: (1) развитие творчества; (2) приобретение и развитие умений коллективной работы.

Продолжительность: 36 часов.

Введённый материал и пояснения

Scratch

Элементы интерфейса: меню FILE / ИМПОРТ ПРОЕКТА…

Группы блоков:

операторы: `повторять до < >.`

Обратите внимание

На выполнение итоговых проектов отводится довольно много времени. Это связано с тем, что итоговый проект — это возможность для творческой самореализации школьника. Кроме того, если ранее не все дети принимали участие в совместной работе над проектом, то при создании итогового проекта все школьники объединяются в мини-группы по 2–3 человека. Распределение ролей в таких группах может быть по принципу «художник — программист — звукорежиссёр — генератор идей» и т. п.

Сложный проект следует разбить на подзадачи и каждому поручить выполнение какой-либо части. В таком случае было бы хорошо предоставить каждому ребёнку, занятому решением своей подзадачи, отдельный компьютер. Такие подпроекты затем «сливаются» в один общий проект. При этом связи между частями проекта должны быть установлены на этапе его планирования. Это так называемые «входные и выходные параметры», описывающие интерфейсы взаимодействия подзадач.

Рассмотрим на более простом примере, как это можно реализовать.

Создать проект «Сажаем сад», в котором Кот сажает дерево или цветок в том месте, где щёлкнули мышкой. Через некоторое время дерево должно вырасти. Размер растения должен зависеть от расстояния до наблюдателя: чем ближе к горизонту, тем растение мельче. На небе са-

жать растения нельзя.

Составим план проекта в виде таблицы объектов и их действий.

Таблица 2

Объект	**Костюмы**	**Действия и взаимодействия объекта**
Сцена	1. Занавес. 2. Основной, раскрашенный так, чтобы небо отделялось от земли	1. При запуске проекта устанавливает фон «Занавес», который растворяется. 2. Передаёт сообщение «Начинай!». 3. Получив «Начинай!», устанавливает основной фон. 4. Считывает координаты указателя мышки и присваивает их глобальным переменным (x, y). 5. Если *y*_координата принадлежит «земле», то шлёт Коту сообщение «Сажай!», иначе — «Не сажай!»
Кот	Два костюма для реализации движения	1. При запуске проекта скрыт. 2. При получении сообщения «Сажай!», перемещается в точку (x,y), после чего передаёт сообщение «Расти!». 3. При получении сообщения «Не сажай!» говорит: «Я не могу посадить дерево на небе!»
Дерево	Несколько костюмов для имитации процесса постепенного роста	1. При запуске проекта спрятано. 2. При получении сообщения «Расти» появляется в координатах (x,y) и «прорастает», оставив штамп. 3. После того, как дерево выросло, передаёт сообщение «Я выросло» и прячется.

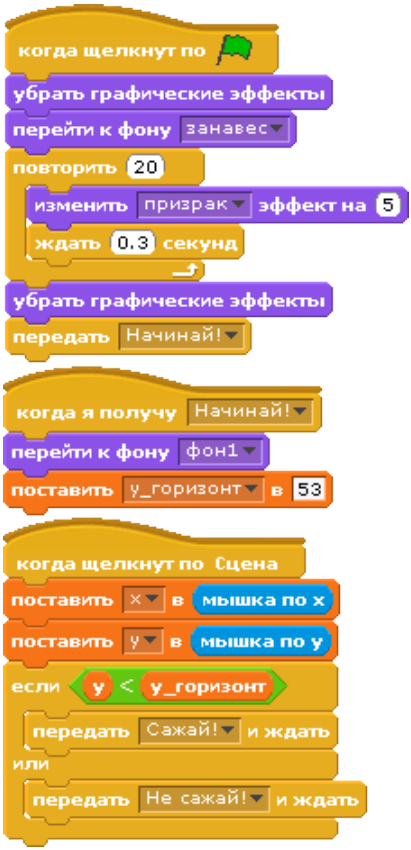
После составления таблицы объектов приступаем к их реализации в отдельных проектах.

Первый проект, который назовём «Сцена», не вызывает сложностей (листинг 23). Здесь *x*, *y* и *y_горизонт* — глобальные переменные. Последней переменной соответствует значение *y*-координаты линии горизонта. Можно не заводить для неё переменную, а использовать само значение. Однако в том случае, когда линия горизонта передвинется (допустим, основной фон перерисовали), придётся изменять это значение в скриптах всех объектов, использующих его.

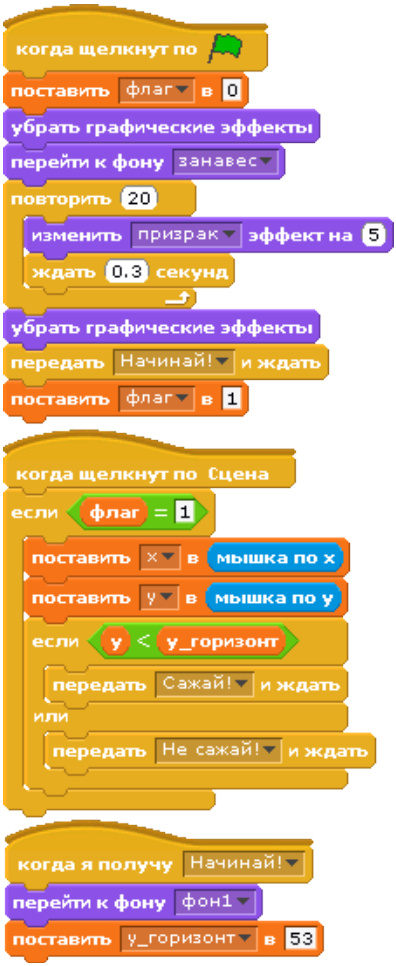
Обратите внимание: в данном варианте, пока исчезает занавес, не исключается возможность щелчка мышкой по Сцене и, соответственно, передачи дальнейших сообщений. Чтобы избежать такой ситуации, можно завести специальную переменную (локальную) — так называемый флаг. Вначале флаг опущен (переменная принимает значение 0). Когда занавес исчезнет, флаг поднимается (переменная принимает значение 1). Сцена должна реагировать на щелчки мышки, только предварительно проверив, поднят ли флаг (листинг 24). Это не очень сложное понятие, однако, если ребята маленькие, можно обойтись и без флага, просто нужно спокойно дождаться, пока исчезнет занавес, и только потом щёлкать мышкой по экрану.

Второй проект касается Кота (назовём проект «Кот»). Самый очевидный вариант перемещения Кота в точку на экране, по которой щёлкнули мышкой — это использовать команды `повернуться к [указатель мыши]` и `плыть [ ] секунд в точку x[ ] y[ ].` Однако, как показывает практика, если сдвинуть указатель мыши в другое положение, Кот, повернувшись за мышкой, может пойти «спиной». Здесь можно использовать тот же приём, что и в проекте с пчелой (проект 15). Создаётся промежуточный объект, например Лопата, со следующими свойствами:

1. При запуске проекта скрыт.
2. При получении сообщения «Сажай!» идёт в координаты (x,y), появляется и передаёт сообщение «Иди, Кот!».
3. Прячется после того, как дерево выросло.

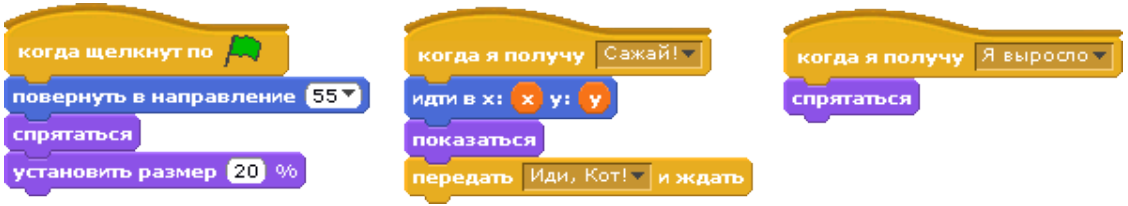


Листинг 23. Скрипты Сцены



Листинг 24. Скрипты Сцены с флагом

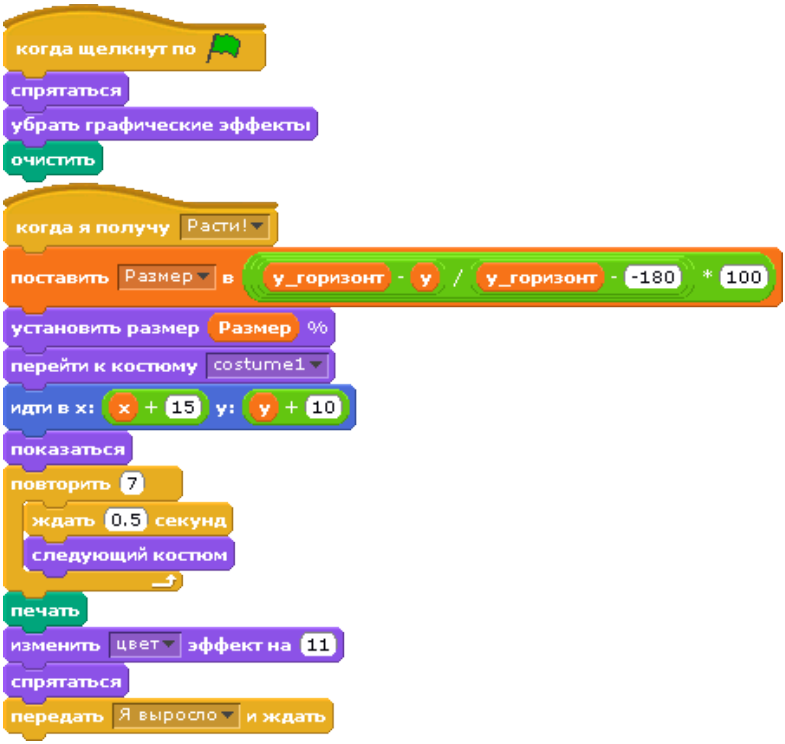
Тогда Кот, получив сообщение «Иди, Кот!», поворачивается к Лопате и перемещается в точку (x,y) (листинг 25, 26). Обратите внимание, Кот не всё время должен двигать ногами, а только тогда, когда перемещается. Для этого использован новый блок `повторять до < >`, который будет выполнять своё содержимое до тех пор, пока ни случится событие, описанное в заголовке цикла (в угловых скобках). В этом отличие от цикла `всегда, если < >`.



Листинг 25. Скрипты Лопаты

Листинг 26. Скрипты Кота

Сложности третьего проекта (назовём его «Дерево») связаны с вычислением размера дерева в зависимости от его удалённости от линии горизонта. Это довольно простая формула, которая, однако, в Scratch выглядит несколько громоздко: расстояние от горизонта до дерева делится на расстояние от горизонта до нижней точки экрана и умножается на 100 % (листинг 27). Размер — локальная переменная. В нашем проекте Дерево имеет 8 костюмов, поэтому цикл смены костюма выполняется 7 раз.



Листинг 27. Скрипты Дерева

Теперь объединим все три проекта в один общий проект.

1. Открыть проект «Сцена».
2. Выбрать пункт меню FILE / ИМПОРТ ПРОЕКТА…
3. В диалоговом окне выбрать проект «Кот».

4. Аналогично выполнить импорт проекта «Дерево».
5. Сохранить проект под новым именем «Сад».

Запуск программы показывает корректность работы программы, при условии, что соответствующие сообщения идентичны.

Публичная защита проектов

Цели: (1) развитие коммуникативных умений; (2) развитие умений публичных презентаций результатов деятельности.

Продолжительность: 12 часов.

Введённый материал и пояснения

Обратите внимание

При оценивании проекта следует обращать внимание на такие элементы проекта, как:

1. наличие заставки и титров с указанием авторства;
2. наличие соответствующего музыкального сопровождения с указанием в титрах авторов музыки;
3. продуманность интерфейса игры;
4. наличие этапа подведения итогов игры;
5. художественное оформление;
6. техническую сложность;
7. защиту от ошибок;
8. практическую значимость проекта.

Резерв учебного времени

Если придерживаться предложенного плана занятий, то резерв учебного времени на два года (2-3 классы) составит 22 часа. Это время может быть использовано учителем как на отдельные занятия, так и на свободное проектирование.

2.2. Рекомендации по выбору межпредметных тем проектной научно-познавательной деятельности школьника

Темы межпредметной проектной деятельности школьника основаны на школьных дисциплинах и зависят от возраста школьника.

Особенности проектной научно-познавательной деятельности младшего школьника

Научно-познавательная деятельность младшего школьника на самых начальных этапах может носить характер только репродуктивной проектной деятельности, что связано, прежде всего, с недостаточной сформированностью его психологических функций и когнитивной сферы, а также с необходимостью изучения новой среды Scratch. Предлагаемые темы проектной деятельности должны быть не трудными и предполагать завершение работы над проектом за одно занятие. В дальнейшем возможно усложнение тем разрабатываемых проектов, которые также должны завершаться за разумное время (несколько занятий). Это обусловлено ситуативным интересом и наличием тенденций достижения. Темы проектов могут быть предложены и детьми. Но при этом особенно важно, чтобы учитель путём наводящих вопросов помог школьнику чётко определиться с конечной целью проекта, составить план проектной деятельности на бумаге и в случае необходимости следил за её продвижением. Следует по возможности сдерживать детскую фантазию (в разумных пределах) с тем, чтобы, с одной стороны, избежать сильной поддержки при выборе им цели, не соизмеримой с собственными возможностями, а с другой — не допустить ситуации, когда цель деятельности не достигнута.

Возможна (но не обязательна) работа в малой группе (2 человека). При этом желательно, чтобы группы формировались на основе личных симпатий. Но учителю необходимо понаблюдать за такой группой, чтобы в случае необходимости

разделить детей, которые не могут сработаться (как показывает наш опыт, чаще всего в таких «неудачных» группах один ребёнок подавляет другого).

После того, как основные блоки Scratch будут изучены (к 4 классу), можно предложить детям попробовать свои силы в междисциплинарных проектах (в начальной школе проект рассчитан на связь двух дисциплин — информатики и какой-либо другой предметной области).

Примерные темы междисциплинарных проектов в начальной школе

1. *«Орфограммы»*. Создание карточек с орфограммами по русскому языку. Проект предполагает разработку нескольких слайдов со словами, у которых орфограммы закрыты кружочками. При щелчке мыши на кружочке он исчезает и открывает орфограмму. Смена слайдов может происходить по нажатию клавиши «пробел» (лёгкий вариант) или с помощью щелчка мыши по отдельно разработанным элементам интерфейса (более трудный вариант). Может быть добавлено музыкальное сопровождение. Готовый проект можно использовать на уроке русского языка в качестве контроля усвоения правописания заданных орфограмм (автор проекта Святослав Дженжер, г. Оренбург, гимназия № 3, адрес проекта http://scratch.mit.edu/projects/sdjenjer/615903).
2. *«Учу английский язык»*. Проект предполагает создание массива английских слов и их перевода. Английские слова и соответствующие им значения на русском языке падают (или летают) случайным образом под музыкальное сопровождение. Необходимо с помощью мыши подобрать правильные пары слов. Если пара составлена верно, то начисляется одно очко и издаётся приветственный звук. Если произошла ошибка, то раздаётся звук разбившегося стекла и очко вычитается. Возможно использование проекта на уроке английского языка (автор проекта Святослав Дженжер,

г. Оренбург, гимназия № 3, адрес проекта http://scratch.mit.edu/projects/sdjenjer/816584).

3. *«Я в тропиках»*. Проект предполагает предварительное изучение некоторой темы дисциплины «Окружающий мир». Требуется написать сочинение «Я в тропиках», «Я в тундре» и т. д. и создать анимированный фильм по сочинению (автор проекта Святослав Дженжер, г. Оренбург, гимназия № 3, адрес проекта http://scratch.mit.edu/projects/sdjenjer/1007492).
4. Игровой проект *«По грибы»* (дисциплина «Окружающий мир»). Предварительно необходимо подготовить (отсканировать или найти в Интернете) изображения съедобных и несъедобных грибов. Грибы случайным образом располагаются на поле. Мышка собирает грибы (можно за время). За съедобные грибы баллы начисляются, а за несъедобные — вычитаются (автор проекта Юля Сальменова, г. Оренбург, гимназия № 3, адрес проекта http://scratch.mit.edu/projects/Jsalmenova/818700).
5. *«Скалолаз»* (тренажёр по решению текстовых задач по математике). На экране появляется текст задачи и ожидание ввода ответа. Бежит таймер. После того, как ответ введён, скалолаз поднимается на один «шаг» вверх, если ответ верный и опускается вниз, если ответ неверный или вышло время. Ведётся подсчёт очков (идея Руслана Шарыпова, г. Оренбург, гимназия № 3).
6. *Тест на общие знания из разных дисциплин*. На экране появляется текст вопроса или задачи и варианты ответа. Реализация может быть различной: более слабые школьники могут остановиться на последовательной подаче вопросов и выборе ответов при помощи мыши. Более сильные школьники могут использовать ввод с клавиатуры и случайную подачу вопросов. Ведётся подсчёт правильных ответов. В конце на экран выводится сообщение о проценте правильных ответов (идея Алёны Пюра и Насти Сальниковой, г. Оренбург, гимназия № 3).

Особенности проектной научно-познавательной деятельности подростка

В 5–8 классах проектная деятельность может иметь большую познавательную направленность, что связано с более устойчивой мотивацией, развитыми когнитивной, волевой и рефлексивной сферами. Длительность работы над проектами постепенно увеличивается и может занимать от нескольких занятий до одной-двух четвертей. Темы проектной деятельности подросток выбирает самостоятельно. Поскольку ведущим видом деятельности подростка является общение, то целесообразно организовать работу в группах по два-три человека и при необходимости оказать помощь в распределении ролей и функций в группе. Если со средой Scratch подростки не знакомы, то можно для начала предложить им три-четыре элементарных проекта. Однако для того, чтобы интерес к среде не пропал, в дальнейшем необходимо усложнение тем проектов с обязательным включением междисциплинарных связей.

Примерные темы междисциплинарных проектов

1. *Создание динамического атласа по истории.* Проект предполагает разработку динамических карт, связанных с определёнными историческими событиями (например, отражающими период взаимоотношений Золотой Орды и Древней Руси).
2. *Изучение звуковых колебаний.* Проект предполагает создание осциллографа, отображающего звуковые колебания. В качестве развития проекта можно строить цифро-аналоговые и аналого-цифровые преобразователи.
3. *Преобразование графиков функций.* Проект предполагает построение графиков и изучение их преобразований при изменении параметров.

Особенности проектной научно-познавательной деятельности старшего школьника

В старших классах внеучебная проектная деятельность уже может носить характер научного исследования и охватывать несколько предметов. В особенности это связано с выбором сферы будущей профессиональной деятельности старшеклассника. Проектное исследование может длиться весь учебный год, но с обязательной публичной демонстрацией окончательных результатов.

Примерные темы междисциплинарных проектов

1. *Изучение геометрических и алгебраических фракталов.* (Математическая теория хаоса)
2. *Моделирование полёта воздушного шара*. (Физика)
3. *Моделирование оптических систем.* (Физика)
4. *Теория химических реакций. Таблица Менделеева.* (Химия)
5. *Построение моделей из молекулярно-кинетической теории газов*. (Физика)
6. *Модель Мальтуса. Модель Лотки-Вольтерра*. (Биология, Экология, Математическое моделирование).

Список литературы

1. Асмолов А. Г., Ягодин Г. А. Образование как расширение возможностей развития личности (от диагностики отбора — к диагностике развития) // Вопросы психологии. 1992. № 1–2. С. 6–13.
2. Базисный учебный план общеобразовательных учреждений Российской Федерации. М.: Просвещение, 2008. 25 с. (Стандарты второго поколения).
3. Герасимова Т. Б. Организация проектной деятельности в школе. // Преподавание истории в школе. 2007. № 5. С. 17–21.
4. Концепция федеральных государственных образовательных стандартов общего образования: проект / Рос. акад. образования; под ред. А. М. Кондакова, А. А. Кузнецова. М.: Просвещение, 2008. 39 с. (Стандарты второго поколения).
5. Краля Н. А. Метод учебных проектов как средство активизации учебной деятельности учащихся: Учебно-методическое пособие / Под ред. Ю. П. Дубенского. Омск: Изд-во ОмГУ, 2005. 59 с.
6. Матвеева Н. В. Информатика и ИКТ. 3 класс: методическое пособие / Н. В. Матвеева, Е. Н. Челак, Н. К. Конопатова, Л. П. Панкратова. М.: БИНОМ. Лаборатория знаний, 2009. 420 с.
7. Матяш Н. В. Психология проектной деятельности школьников в условиях технологического образования/ Под ред. В. В. Рубцова. Мозырь: РИФ «Белый ветер», 2000. 285 с.
8. Патаракин Е. Д. Учимся готовить в среде Скретч (Учебно-методическое пособие). М: Интуит.ру, 2008. 61 с.
9. Пахомова Н. Ю. Метод учебного проекта в образовательном учреждении: Пособие для учителей и студентов педагогических вузов. М.: Аркти, 2008. 112 с.
10. Примерные программы начального общего образования [Электронный ресурс] // Федеральный государственный

образовательный стандарт [сайт]. URL: http://standart.edu.ru/catalog.aspx?CatalogId=531
11. Скретч [Электронный ресурс] // Материал с Wiki-ресурса Letopisi.Ru — «Время вернуться домой». URL: http://letopisi.ru/index.php/Скретч
12. Хохлова М. В. Проектно-преобразовательная деятельность младших школьников. // Педагогика. 2004. № 5. С. 51–56.
13. Цукерман Г. А. Что развивает и чего не развивает учебная деятельность младших школьников? // Вопросы психологии. 1998. № 5. С. 68–81.
14. Школа Scratch [Электронный ресурс] // Материал с Wiki-ресурса Letopisi.Ru — «Время вернуться домой». URL: http://letopisi.ru/index.php/Школа_Scratch
15. Scratch | Home | imagine, pgogram, share [сайт]. URL: http://scratch.mit.edu
16. Scratch | Галерея | Gymnasium №3 [сайт]. URL: http://scratch.mit.edu/galleries/view/54042

Приложение 1

Раздаточный материал к музыкальным проектам

Таблица 3

Обозначение длительности нот и пауз

Обозначение		Длительность		Количество нот в одном музыкальном такте		
нота	пауза	в музыке	в Scratch	$\frac{2}{4}$	$\frac{3}{4}$	$\frac{4}{4}$
		целая	1	—	—	
		половина	0,5			
		четверть	0,25			
		три восьмых	0,325			
		восьмая	0,125			

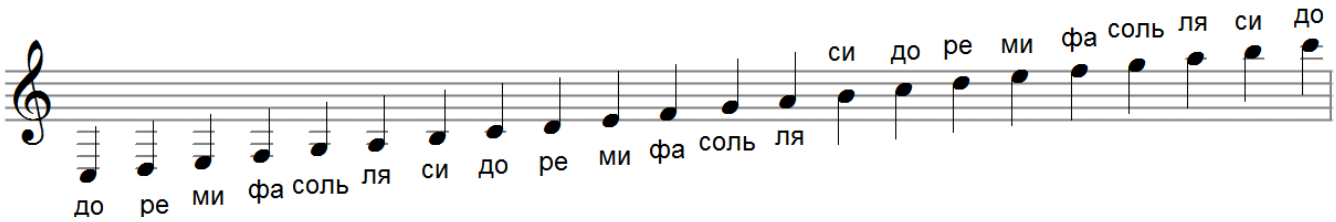


Рис. 9. Гамма ДО мажор

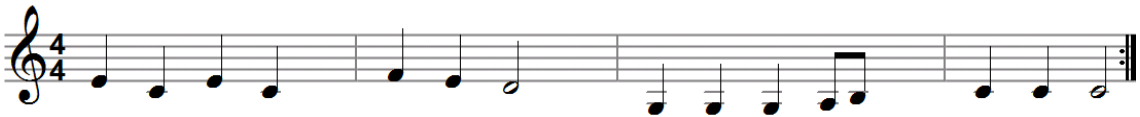


Рис. 10. Песенка «Чижик-пыжик»

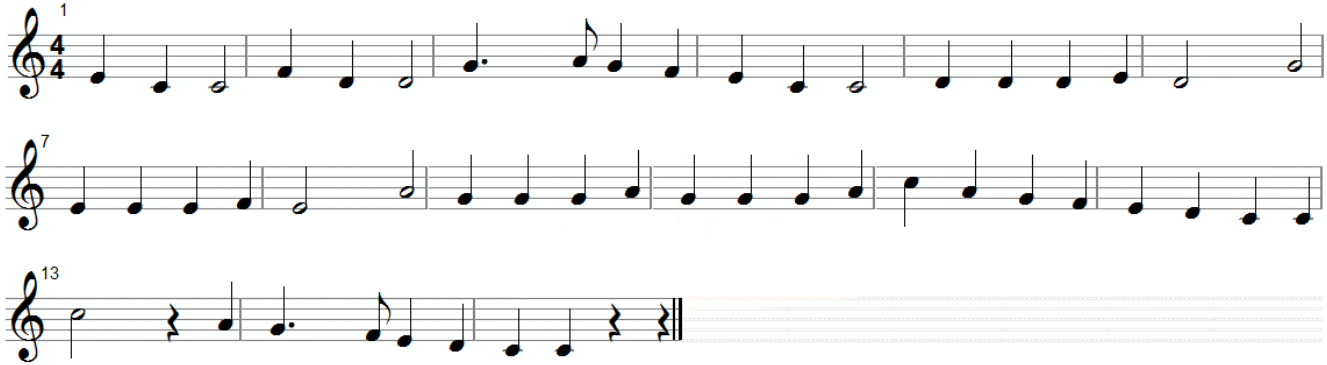


Рис. 11. Основная партия песенки «Тра-та-та»

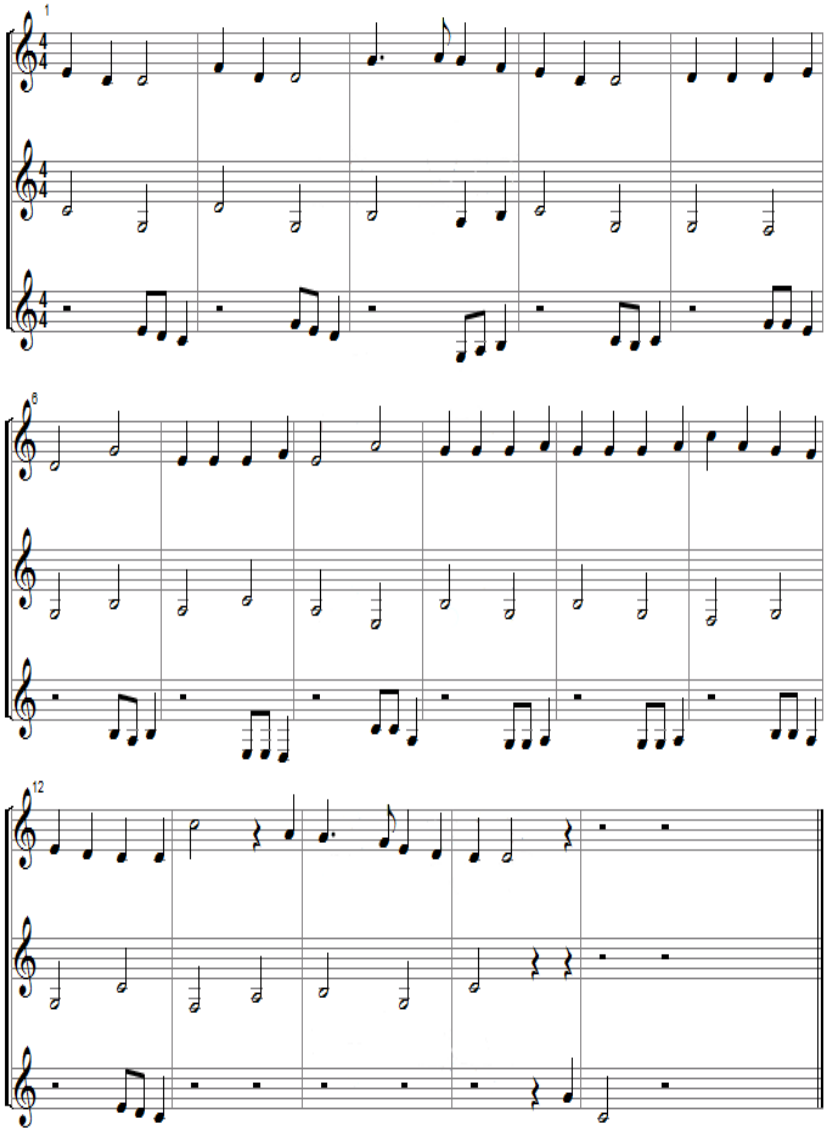


Рис. 12. Вариация песенки «Тра-та-та» для трёх инструментов

Приложение 2

Листинг 28. Синхронизированное многоголосье (проект 10)

Рис. 13. Скрипт основной партии (флейта)

Рис. 14. Скрипт второй партии (акустическая гитара)

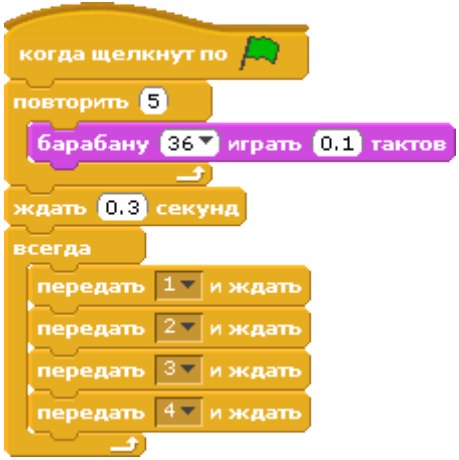


Рис. 15. Скрипт дирижёра

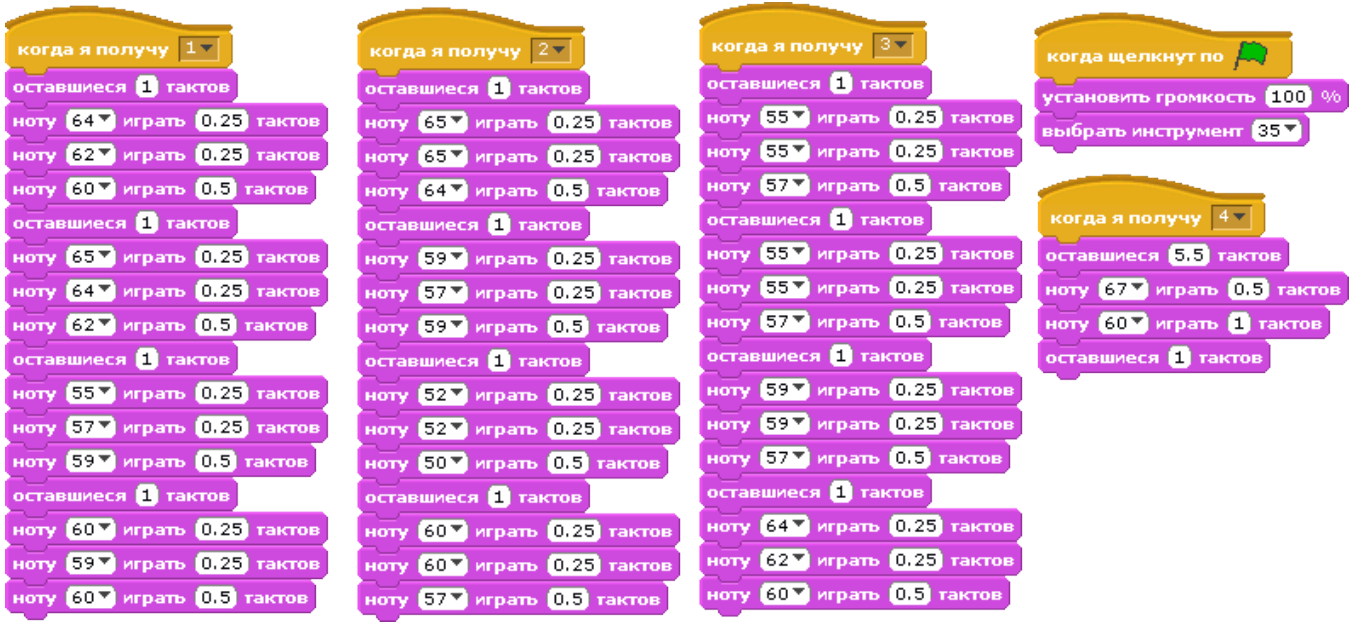


Рис. 16. Скрипт третьей партии (бас-гитара)

Список сокращений

БУП — базисный учебный план

ИИ — искусственный интеллект

ПНПД — проектная научно-познавательная деятельность

ВПНПД — внеучебная проектная научно-познавательная деятельность

ВПНПДШ — внеучебная проектная научно-познавательная деятельность школьника

Указатель терминов

Об авторах

Рындак Валентина Григорьевна

Заслуженный деятель науки РФ, доктор педагогических наук, профессор, директор Института педагогики и менеджмента, заведующая кафедрой общей педагогики Оренбургского государственного педагогического университета
ped@bk.ru

Дженжер Вадим Олегович

Кандидат физико-математических наук, доцент, заведующий кафедрой информатики и методики преподавания информатики Оренбургского государственного педагогического университета
vdjenjer@yandex.ru

Денисова Людмила Викторовна

Кандидат педагогических наук, доцент кафедры информатики и методики преподавания информатики Оренбургского государственного педагогического университета
lv-denisova@yandex.ru

Рындак Валентина Григорьевна
Дженжер Вадим Олегович
Денисова Людмила Викторовна

ПРОЕКТНАЯ ДЕЯТЕЛЬНОСТЬ ШКОЛЬНИКА В СРЕДЕ ПРОГРАММИРОВАНИЯ SCRATCH

Учебно-методическое пособие

Подписано в печать 20.12.09. Формат 60x84 $^{1}/_{16}$.
Бумага офсетная. Гарнитура «Times».
Объём 3,05 уч.-изд. л., 6,16 усл. печ. л.
Тираж 100 экз. Заказ № 095.

Отпечатано в типографии ГОУВПО «ОГИМ»
460038, г. Оренбург, ул. Волгоградская, д. 16.
Тел./факс: (3532) 305-000