Учимся готовить в среде Скретч

Придумай – Запрограммируй – Поделись

Версия 2.0

Евгений Патаракин

Введение 3
Что можно делать и чему можно научиться со Скретчем? 7
Наш первый сценарий 9
Где что лежит.....13
Создание новых объектов.....15
Управляющие программы - скрипты.....17
Содержимое ящиков.....19
Внешний вид объекта (фиолетовый ящик) 19
Синий ящик - Движение.....21
Команды рисования (темно-зеленый ящик) 23
Кирпичики чисел (зеленый ящик) 24
Кирпичики контроля (желтый ящик).....25
Датчики (голубой ящик) 27
Лиловый ящик - Звуки.....28
Переменные (оранжевый ящик) 29
Примеры рецептов в среде Скретч.....30
Рецепт № 1 Мультипликация.....30
Рецепт № 2 Взаимодействие объектов – кошка убегает от мышки.....33
Рецепт № 3 Играем с геометрическими фигурами.....35
Рецепт № 4. Играем с буквами.....37
Рецепт № 5. Поиграем со случайными надписями на экране 41
Рецепт № 6 Модель битвы драконов и демонов.....42
Рецепт № 7 Управление через обмен сообщениями.....46
Рецепт № 8 Сказка про Зайца и Лису.....48
Рецепт № 9Сказка про репку.....50
Рецепт № 10 Скретч-квест.....52
Сообщество Скретч.....56
Как вступить в сообщество Скретч?.....59
Как просматривать, скачивать и видоизменять проекты сообщества?.....60
Как можно пополнить архивы сообщества Скретч своими проектами?.....62
Заключение – что после Скретча?.....65
Squeak.....65
Alice.....66
Крокет.....67
NetLogo.....68
Скретч и веб 2.0.....70
Литература.....72

Введение

Скретч (Scratch) это новая среда программирования, которая позволяет детям создавать собственные анимированные интерактивные истории, игры и модели. Этими произведениями можно обмениваться с сообществом автором Скретч-проектов по всему миру. В Скретче можно играть с различными объектами, видоизменять их вид, перемещать их по экрану, устанавливать формы взаимодействия между объектами. Это объектно-ориентированная среда, в которой блоки программ собираются из разноцветных кирпичиков команд.

В названии языка скрывается отсылки к нескольким метафорам:

1. Scratch ─ Царапина, скрип, перо, каракуля, старт
2. Old Scratch – (шутл.) – Сатана, дьявол
3. Scratch-cat – злючка
4. Scratch paper – черновик
5. To start from the Scratch – начинать с нуля, на пустом месте

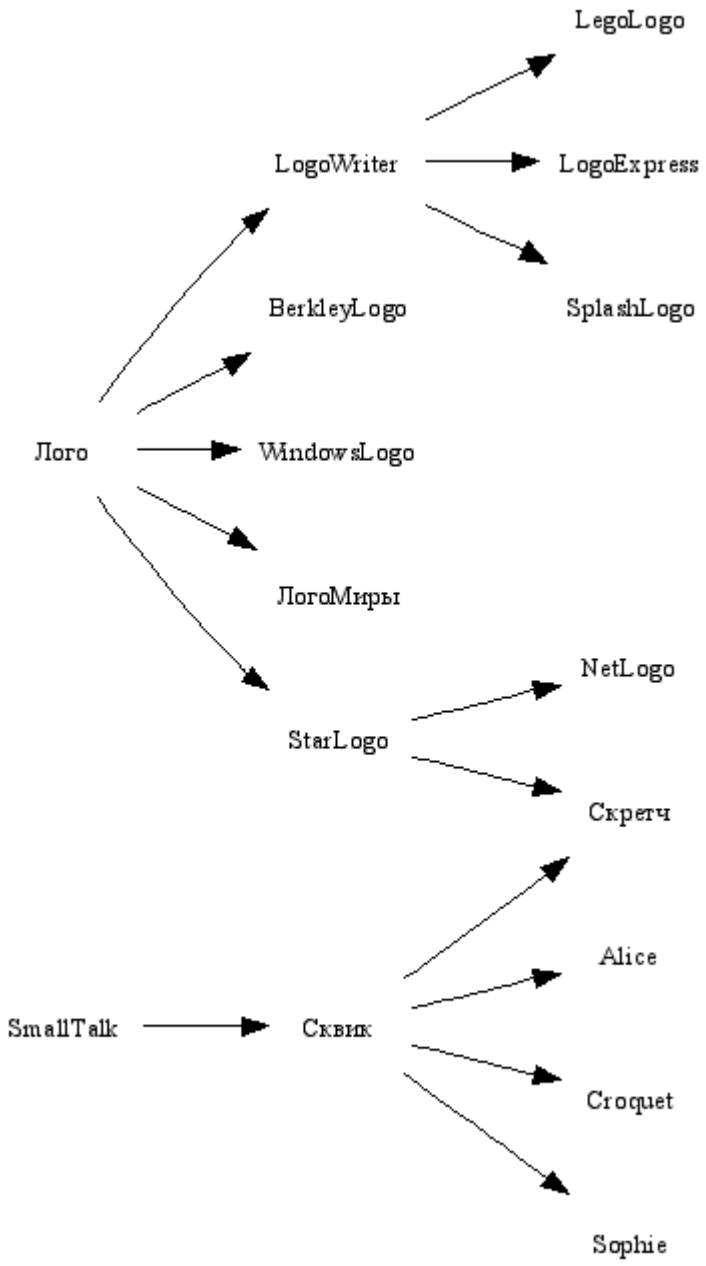
Мы можем начинать пользоваться языком с нуля, не обладая никакими предварительными знаниями о программировании. Важно отметить, что Скретч приходит в современный мир вместе с другими важными педагогическими инициативами. Это движение свободных программ и движение «Один ребенок – один компьютер» - «1:1». Согласно идеологии этого движения ребенок должен осваивать не программы-приложения, а различные способы деятельности: создавать свои собственные истории, придумывать игры, разрабатывать компьютерные модели. Для таких целей Скретч подходит как нельзя лучше. Разработчики языка думают о том, как дети в среде Скретч смогут самостоятельно осваивать современную культуру, играть с образами, звуками, мультипликацией. В этой среде ученики не используют готовые компьютерные игры, а конструируют свои собственные игры, истории и модели. В ходе этой созидательной

деятельности у учеников формируется свобода обращения с различными элементами окружающей медиакультуры.

Цель движения «1:1» состоит в том, чтобы не просто дать детям технические средства, но помочь им научиться думать лучше и глубже, чем думают большинство взрослых. Для достижения этой цели создаются новые технические и программные среды – как новые формы электронной бумаги, на которой могут быть представлены новые пути представления новых мощных идей. Персональные компьютеры, доступные для каждого ученика, создают новую среду чтения и письма, в которой люди могут играть с новыми мощными идеями совсем не так, как они это делали с книгами. Образцы этой новой литературы публикуются в сети Интернет, для того, чтобы и взрослые и дети могли бы читать, играть с ними, классифицировать, обсуждать и критиковать. Новый стиль усвоения знаний дает ученикам возможность обдумывать идеи и представлять их в сетевой форме доступной для обсуждения и критики. Как отмечает Алан Кей «возможность обсуждения и критики является одним из самых древних оснований обучения». Опыт взаимодействия с глубокими идеями помогает вырастить людей, которых будет значительно труднее ввести в заблуждение, поскольку у них будет заложен опыт критического отношения и активной работы с любыми мифами и мимами.

Впервые отношение к компьютеру как к дружественному средству, которое ученик может контролировать и при помощи которого он может думать новыми, более эффективными способами, сформулировал Сеймур Пейперт. В своих работах он сместил направление педагогических инноваций с поиска лучших методов преподавания на поиск *лучших объектов*, при помощи которых можно конструктивно действовать и размышлять о своей деятельности. Пейперт и его коллеги предположили, что построение собственных интеллектуальных структур осуществляется учеником наиболее эффективно в том случае, если он вовлечен в создание реального конечного продукта: замка из песка, машины, книги или компьютерной программы. Лучше всего обучение происходит в том случае, если ученик вовлечен в деятельность, которая сопровождается размышлением и обсуждением о том, что он делает. Люди учатся значительно эффективнее, если они вовлечены в создание объектов, которые имеют для них личное значение. Компьютеры и компьютерные сети позволяют людям получать доступ к новым объектам, создавать и играть с гораздо большим числом искусственных объектов, чем это было возможно раньше. При помощи этих новых учебных объектов ученики могут конструировать новые истории, новые произведения и объекты. Они могут проводить исследования этих новых объектов и описывать их поведение.

В книге «Переворот в сознании» Пейперт писал о необходимости «*объекта, при помощи которого можно лучше думать*». В поведении такого объекта ученик может увидеть свое собственное поведение. Впервые этот подход был воплощен в языке Лого, когда ученик получил в свое распоряжение агента - Черепашку, которой можно было давать простые команды: вперед, назад, направо, налево, повтори и т.д. Из этих простых кирпичиков постепенно складываются значительно более сложные микромиры. Эволюция Лого и возникновение языка Скретч представлены на следующей схеме:



Уже середине 90-х годов Мич Резник и его группа разработали продукт StarLogo, в котором действовали множество черепашек. Продукт был нацелен именно на изучение закономерностей и феноменов, в которые вовлечено множества агентов. Синтез идей StarLogo и возможностей языка Squeak привели к созданию языка Скретч.

К простым действиям черепашки Лого в языке Скретч добавились мощные идеи и технологии параллельных действий и обмена информацией. Теперь они доступны для образования.

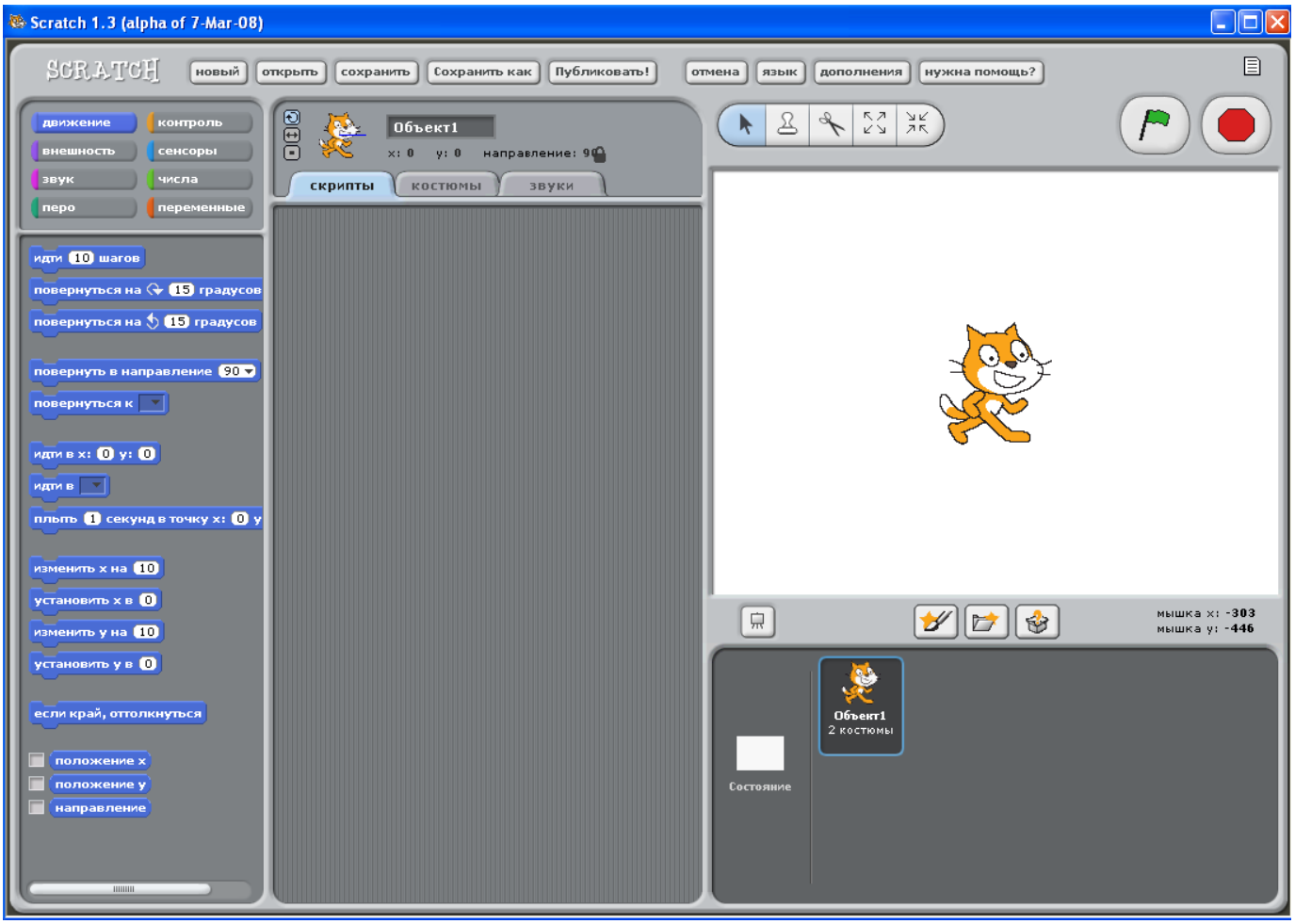
Скретч используется метафора кирпичиков Лего, из которых даже самые маленькие дети могут собрать простейшие конструкции. Дети могут собирать свои программы-процедуры из блоков так же, как они собирали конструкции из разноцветных кирпичиков. Из конструкций и управляющих структур могут быть собраны различные агенты, выполняющие простые инструкции. Эти агенты могут взаимодействовать между собой и воссоздавать сообщество в среде Скретч. В результате выполнения простых команд может складываться сложная модель, в которой будут взаимодействовать множество объектов, наделенных различными свойствами. Начальный уровень программирования столь прост и доступен, что Скретч рассматривается в качестве средства обучения не только старших, но и младших школьников.

Когда ученики создают проекты в Скретче, они осваивают множество навыков 21 века:

- творческое мышление,
- предметное общение,
- системный анализ,
- беглое использование технологий,
- эффективное взаимодействие,
- проектирование,
- постоянное обучение

Что можно делать и чему можно научиться со Скретчем?

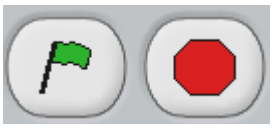
Чтобы начать работать в среде Скретч, достаточно ткнуть указателем мышки в мордочку симпатичному зверьку на экране – Скретч и запустится. Перед нами экран программы Скретч.



Справа большая область, на которой находится главный герой программы. Здесь происходят события, сюда можно добавлять новые программируемые объекты. Белый фон может меняться на другие. И героям и фону, на котором они действуют, можно давать команды. Команды разложены как кирпичики в ящичках в левой верхней части экрана. В данный момент открыт синий ящик, и мы видим команды движения, которые в

этом ящике лежат. Чтобы выполнить команду или собрать из нескольких команд сценарий – скрипт, нужно положить команду в центральную серую область.

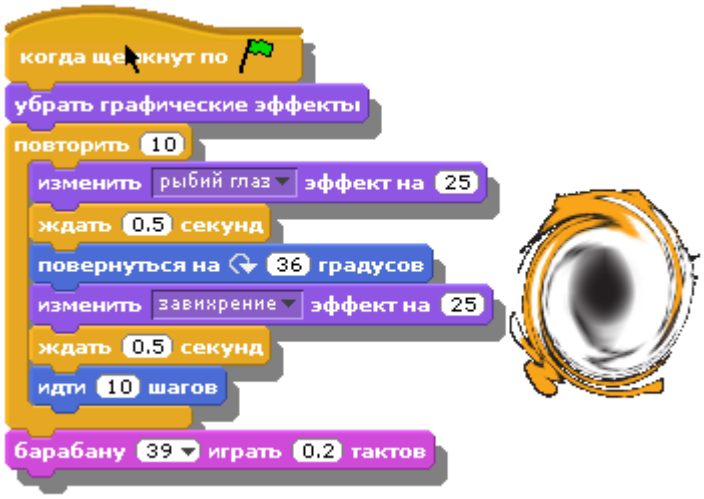
Справа - область экрана или мира, где существует кошка и все другие существа, которых мы можем в мир призвать.



Зеленый флажок служит в качестве сигнала, который воспринимают и на который реагируют все присутствующие в данном проекте объекты Красная кнопка Stop - останавливает движения и действия всех объектов. Опыт и метафоры Лего-Лого в среде Скретч лежат на поверхности.

Мы можем взять любой объект и осуществить над ним действия:

- видоизменить его внешний вид при помощи фиолетовых команд-кирпичиков внешнего вида;
- переместить его при помощи синих команд кирпичиков перемещения;
- поместить команды перемещения и видоизменения внутрь оранжевых блоков управления;
- добавить к управляющим блокам фиолетовые кирпичики звуков;
- и т.д.



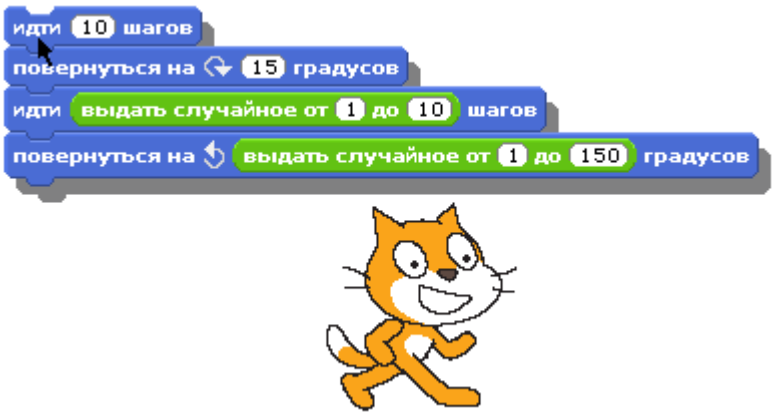
И наш исполнитель 10 раз

(Округлился, повернулся, искривился, прошел 10 шагов),

а после этого хлопнул в ладоши.

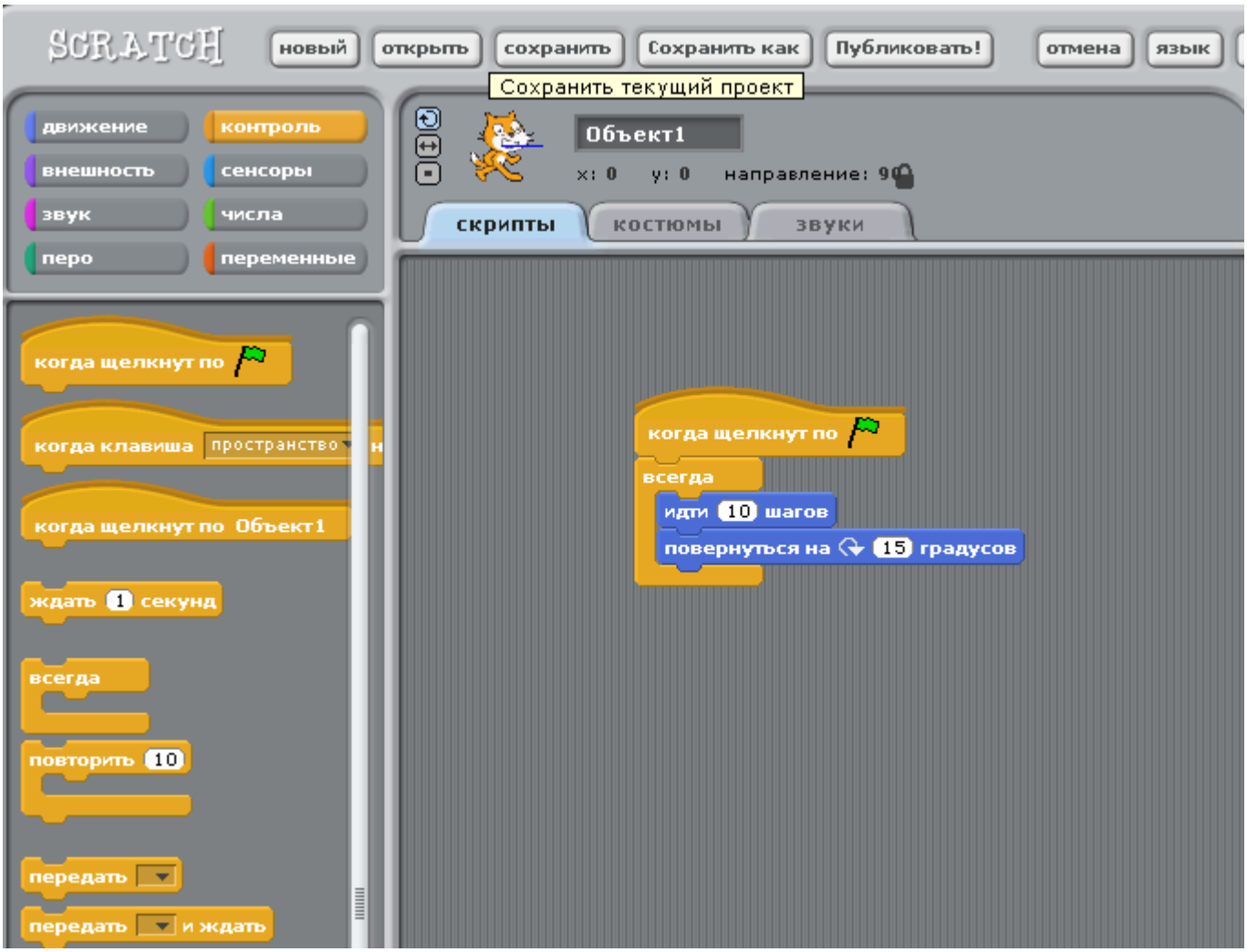
Наш первый сценарий

На следующем рисунке показано, как мы положили 2 команды из синего ящика в центральную часть и слепили из них общий блок. Команды легко соединяются между собой – как кирпичики Лего. Если мы теперь щелкнем по блоку из двух кирпичиков, то котенок выполнит 2 команды сценария: Иди 10 шагов, поверни вправо на 15 градусов. Цифры 10 и 15 уже были написаны внутри кирпичиков, но мы можем их изменять. К двум командам из синего ящика мы добавили еще блок из зеленого ящика с цифрами. Этот блок позволяет получить случайное число в заданном интервале. Например, кошка может сделать какое-то число шагов в интервале от 1 до 10 и повернуть против часовой стрелки на какое-то число градусов.



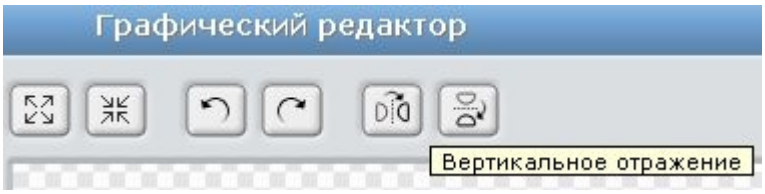
Теперь мы добавим к командам движения из синего ящика парочку команд управления из желтого ящика. В жёлтом ящике лежат команды не очень похожие на остальные кирпичики. Большинство из них устроены так, что в них можно запихивать другие кирпичики и организовывать их циклическое выполнение. Мы сразу возьмем самый страшный блок forever = всегда и запихаем в него наши синие кирпичики движения. Сверху прикрепим блок запуска сценария по зеленому флажку и наш первый скрипт готов. Теперь наш котенок будет вечно вертеться.

Мы можем сохранить результаты своей работы в отдельный проект. Для этого нужно щелкнуть по кнопке – «Сохранить» и записать название своего проекта. К сохраненному проекту всегда можно вернуться и продолжить над ним работу.



Продолжим хождение котенка по экрану и зададим простое и полезное условие: если коснулся края рабочего поля, то не убегай, а оттолкнись и измени направление. Теперь наш котенок ходит из стороны в сторону. Только почему-то так получилось, что когда он идет направо, то все хорошо, а когда налево, то идет вверх ногами.

Попробуем помочь котенку всегда двигаться вверх головой. Для этого нам нужно будет добавить еще один образ или костюм, который может надевать котенок. Сначала делаем операцию «Копировать», а затем в режиме редактирования переворачиваем котенка вверх головой.



В результате вертикального отражения в графическом редакторе мы получаем два образа котенка:

Теперь напишем условия поведения котенка:

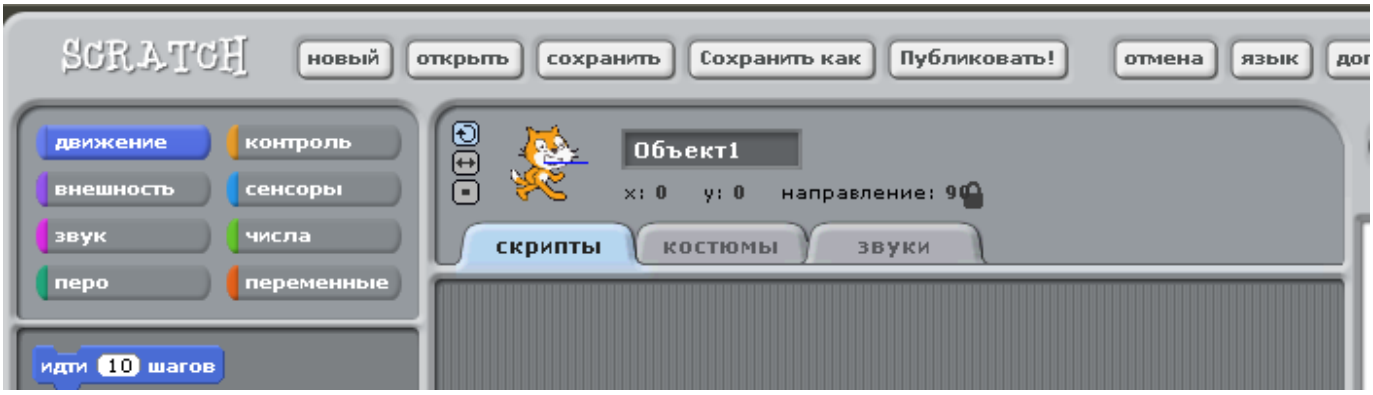
- Если идем налево, то надеваем костюм № 3
- Если идем направо, то надеваем костюм № 1

Можем написать другой вариант, при котором будет проверять не свое направление, а свое положение на экране.

- Если близок к правому краю, надевает костюм № 3 и устанавливает направление -90.
- А если близок к левому краю, то надевает костюм № 1 и устанавливает направление 90.

Где что лежит

Познакомимся подробнее с возможностями, которые нам доступны в меню:



Верхняя часть:

Новый Создать Новый проект

Открыть – открыть проект

Сохранить – сохранить текущий проект

Сохранить как – сохранить проект под новым именем

Публиковать! – размещение проекта в общей сетевой коллекции

Отмена – отмена последней команды.

Язык – выбор интерфейса. Изначально Скретч был написан на английском, но в последние годы добавилось еще множество языков, которые понимают объекты Скретч.

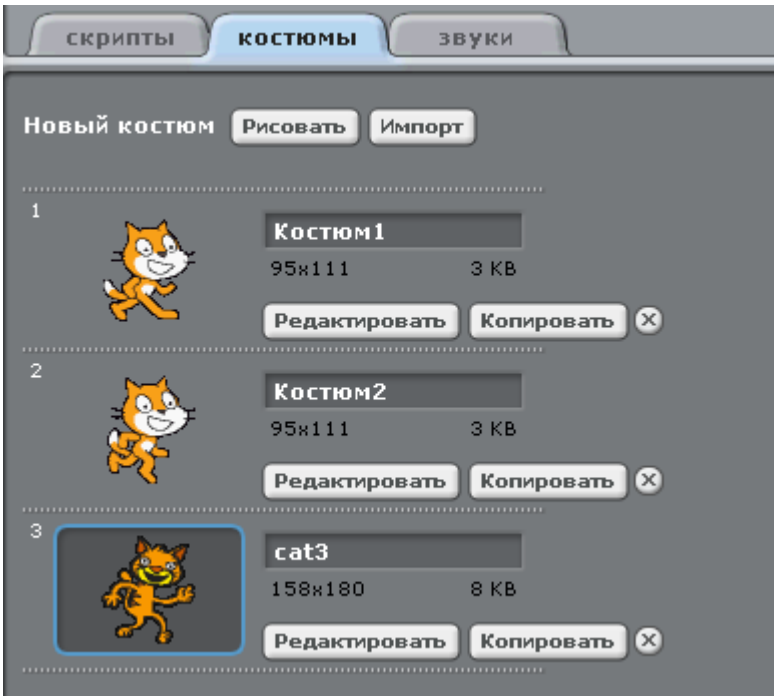
Левая часть рабочего экрана представляет ящички, в которых лежат кирпичики для программирования.

- Фиолетовый **Внешность**- Изменение внешнего вида
- Синий **Движение** - Перемещение объектов
- Лиловый – **Звуки** - звуки, ноты и множество инструментов
- Зеленый – **Числа** Операции с числами, сравнение, логические операторы
- Желтый – **Контроль** - контролирующие операторы
- Ярко-зеленый – **Перо** возможность рисовать на экране
- Голубой – **Сенсоры** – датчики и возможность определять направление, расстояние и т.п.
- Красный - **Переменные** – переменные, которые можно передавать в скрипты и которым можно присваивать значения от датчиков.

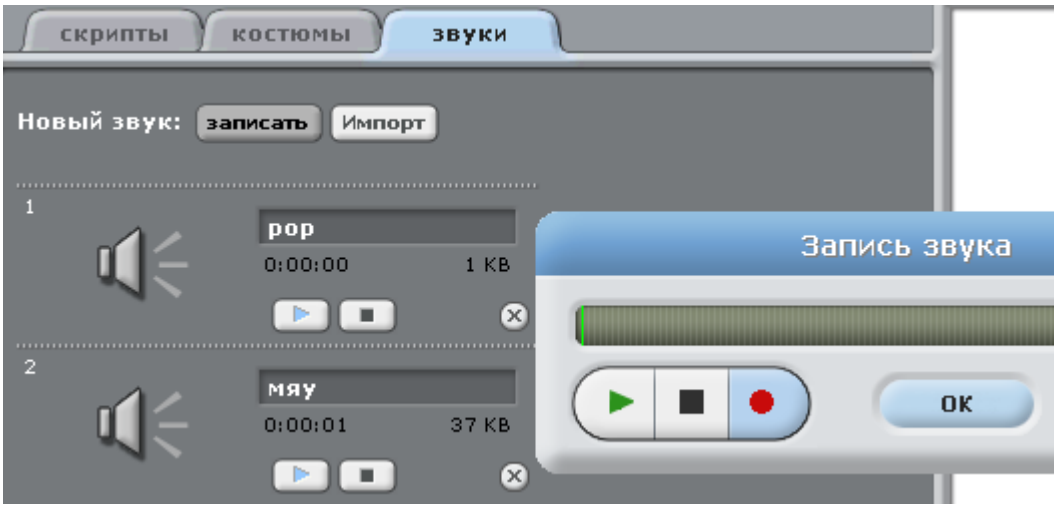
Центральная часть рабочего экрана Скретч.

Скрипты - область, где из элементов собираются управляющие скрипты

Костюмы, которые может надевать на себя кошка или другой герой. Мы можем добавлять костюмы из библиотек или рисовать новые.



Звуки - звуки, которые кошка может издавать.



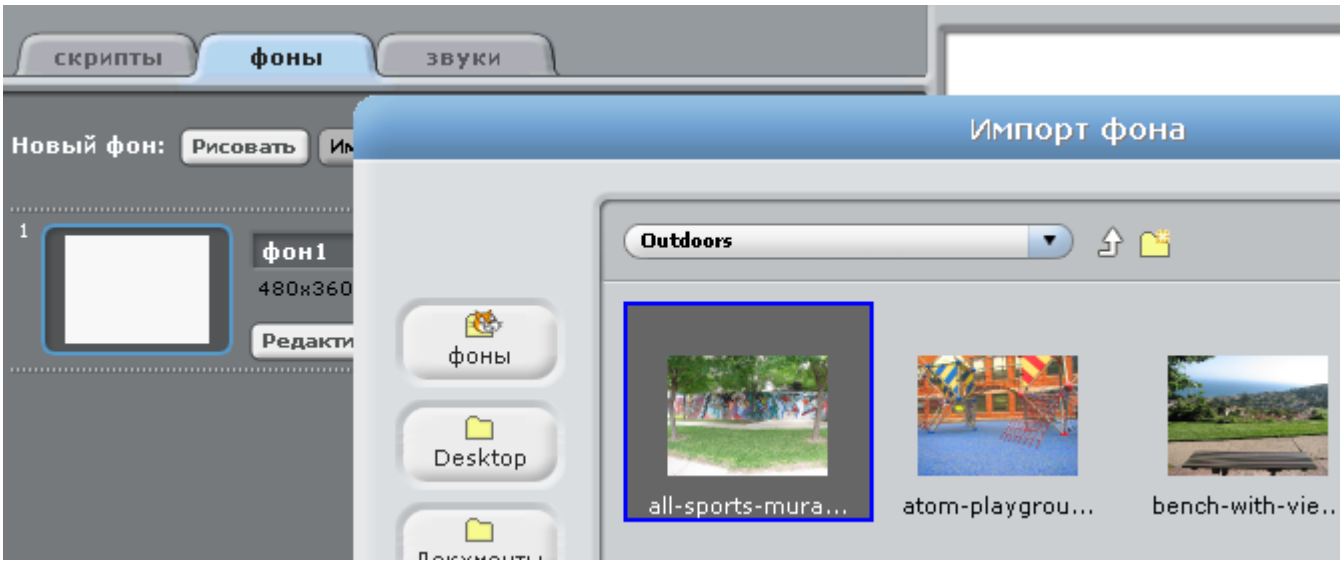
Мы можем добавлять звуки из библиотек или сами записывать новые – в Скретч встроена программа аудиозаписи.

Создание новых объектов

Для того чтобы вызвать новое существо – (новый объект) под рабочей для объектов областью лежат несколько рабочих возможностей:

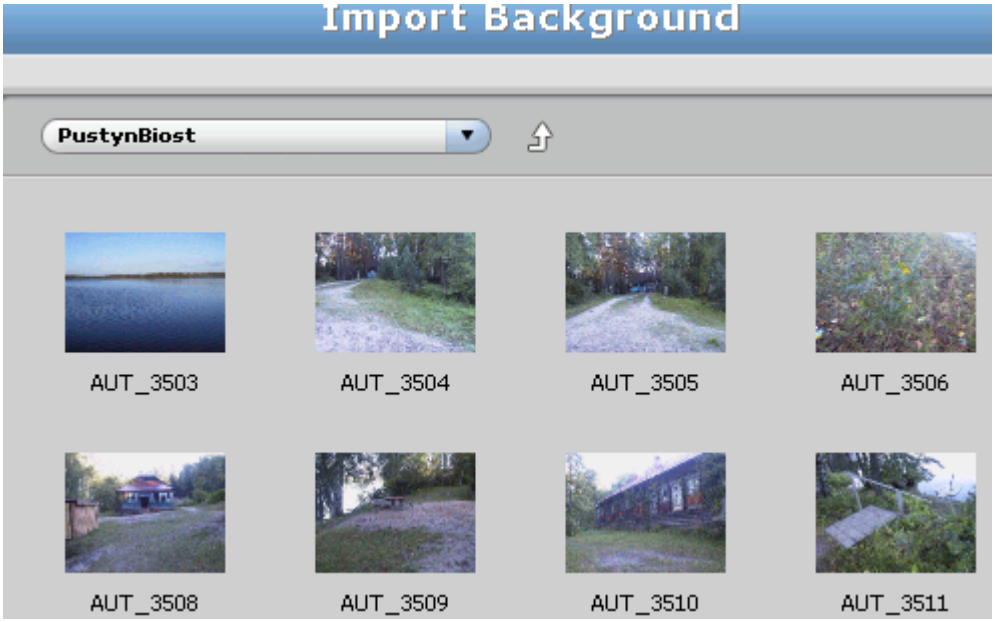
- Кисточка, чтобы нарисовать новый объект
- Папка, чтобы достать готовый объект из библиотеки
- Коробочка со знаком вопроса, откуда можно достать случайный объект.

Среда, на фоне которой происходит действие, является таким же импортируемым и видоизменяемым объектом. Мы можем нарисовать фон, на котором происходит действие. Мы можем взять для этого фона картинку из коллекции, которая пришла к нам вместе со средой Скретч.

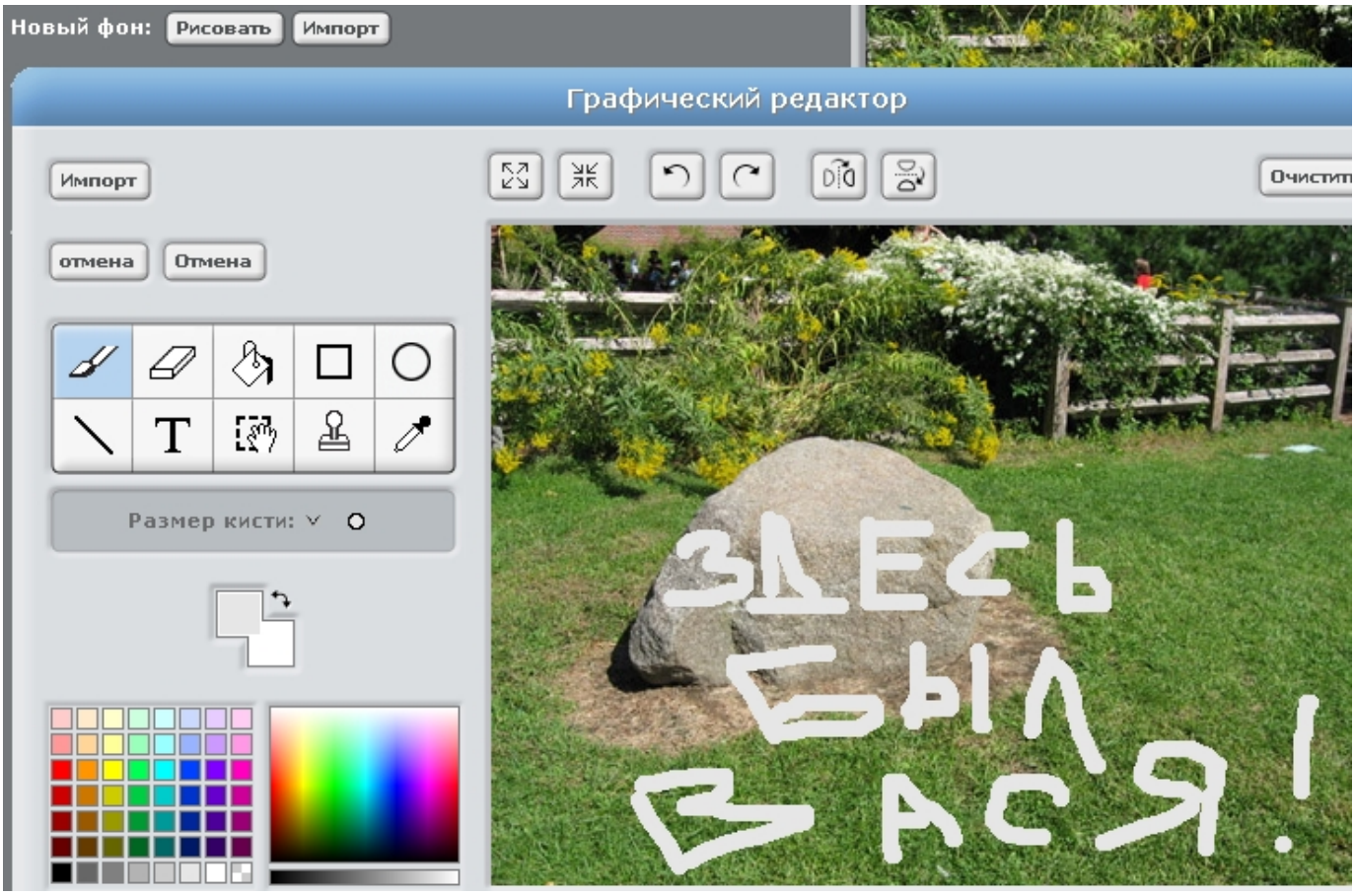


Мы можем самостоятельно добавить к коллекциям фона свои рисунки или фотографии. Размер рисунков и фотографий для фона – 480 на 360

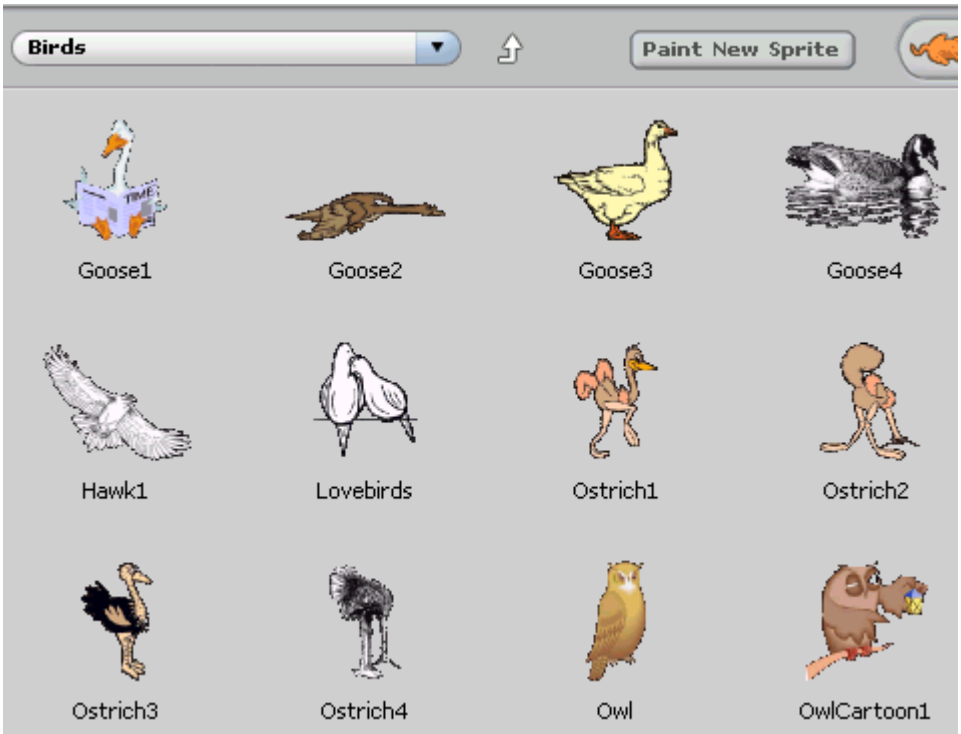
В данном примере мы добавили фотографии, которые были сделаны на биостанции Нижегородского Университета на берегах пустынских озер.



Когда мы создаем новое существо, то мы можем выбрать его внешний вид из библиотеки костюмов - тут люди, звери, растения, продукты, машины, сказочные существа. Мы можем добавить любое изображение и отредактировать это изображения прямо внутри графического редактора среды Скретч.



Мы можем перейти в папку Nature – природа и выбрать там папку, в которой сложены растения - деревья, травы и листья. Другая папка содержит изображения птиц, которые мы и будем использовать в последующих примерах:



Важно отметить, что папки с изображениями объектов мы можем создавать и пополнять самостоятельно.

Управляющие программы - скрипты

Перейдем к ящичкам, в которых сложены кирпичики. Если мы посмотрим на содержимое ящичков, то увидим, что содержимое отличается не только цветом. Есть ящички, где большей частью лежат строительные блоки, которые можно объединять друг с другом. Это кирпичики для изменения внешнего вида объектов, управления звуками, перемещения объектов и рисования. Эти строительные кирпичики делятся на закрытые и самодостаточные, к которым ничего добавить нельзя и внутрь которых ничего нельзя добавить. Например:

Если край, оттолкнуться

Есть кирпичики, в которых можно изменять значения или вставлять в них значения. Например: **Идти(20), Установить размер ()%**

Есть ящички с блоками, которые нельзя соединять с другими, но которые можно вставлять внутрь других строительных блоков. Это ящики, где лежат цифры и датчики.

Есть ящик **Контроля** с управляющими конструкциями, которые позволяют организовывать повторения команд, ставить условия, при которых блоки команд начинают выполняться и условия, когда действие команд прекращается

Команды приобретают смысл, когда Вы объединяете их в последовательность

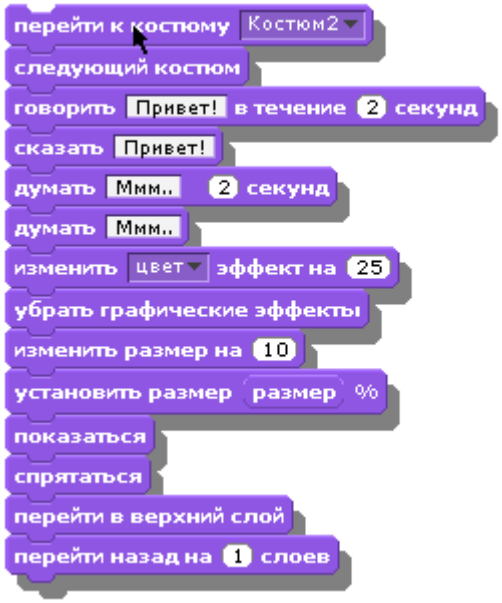
Например:

И птица начинает махать крыльями.

Все команды применимы не только к подвижным объектам, но и фону! Т.е. мало того, что можно добавлять разные фотографии в качестве фона. Над этими фоновыми фотографиями можно совершать любые операции при помощи команд из фиолетового ящика.

Содержимое ящиков

Внешний вид объекта (фиолетовый ящик)



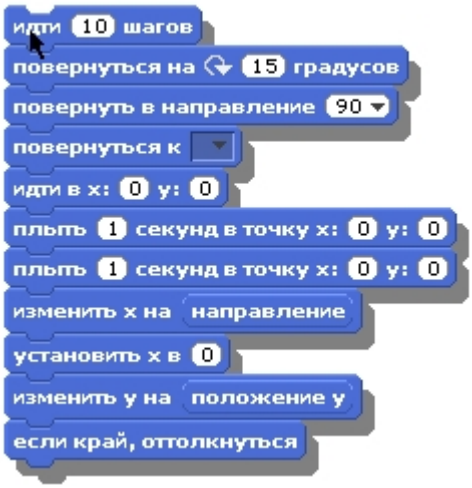
Все команды однозначны. Чаще всего употребляются команды «Изменить()» и «Убрать графические эффекты». Переходы между слоями легче всего пояснить на примере 2-х одинаковых спрайтов котенка. У одного из них покрасим глаза в голубой цвет. Если мы теперь попросим одного из котят менять слой, то получится простая мультипликация – котенок будет менять цвет глаз.

Здесь приведен самый простой пример, но мы можем увеличить число объектов.

Команда	**Назначение**
Перейти к костюму()	Перейти к костюму (можно выбрать костюм героя)
Выбрать костюм()	По умолчанию объект одет в костюм кошки, но мы можем выбрать и другие костюмы из библиотеки, пользуясь командой Импорт.
Изменить значение костюма	Изменить значение костюма на одну или несколько единиц. Имеет смысл только тогда, когда у нас несколько «импортных» костюмов.
Сказать	Сказать фразу, которую мы можем записать в окошечке команды. К сожалению, пока можно использовать только латинские буквы.
Думать	Думать – отличается от «говорить» только графическим оформлением фразы
Убрать графические эффекты	Очистить все графические эффекты. Если мы производили над объектом видоизменение, то в результате этой команды все они отменяются
Изменить()	Изменить (цвет, округлость, яркость) () – видоизменить объект по одному из перечисленных параметров на указанную величину.
Показаться	Показать объект – он становится видимым
Спрятаться	Спрятать объект - он становится невидимым
Перейти в верхний слой	Объект перемещается в первый слой
Перейти назад на () слоев	объект уходит на несколько слоев внутрь
Значение костюма	Возвращает значение костюма, который в данный момент одет на нашем исполнителе.

Синий ящик - Движение

В синем ящике сложены строительные блоки, управляющие перемещением объектов по экрану.



Кошка Скретч является прямым потомком черепашки Лого, и она наследовала многие из команд передвижения. При этом ошибку в команде сделать сложнее, поскольку количество шагов или градусов интегрированы с самой командой. Поверхность экрана, по которой перемещается Кошка, напоминает по своим свойствам экран, по которому перемещалась Черепашка. Центральная точка экрана соответствует координатам X=0, Y=0. Вправо от нулевой точки X возрастает и это область положительных значений. Влево – область отрицательных значений.

Сложнее всего использовать блоки, которые не являются командами, а возвращают значения – положение x, y и направление. Простой пример:

Мы хотим, чтобы котенок последовательно переходил из нижнего правого угла в левый верхний угол.

Команда	Назначение
идти	пройти вперед указанное число шагов. Например: Идти(10) – сделать десять шагов назад Идти (-30) – сделать 30 шагов назад
Повернуться на	повернуться - стрелочка указывает по часовой или против часов стрелки выполняется поворот. Например: повернуться на 45 градусов (стрелочка указывает по или против часовой стрелки)
Повернуться в направление()	Повернуться в указанном направлении. Например: повернуться в направление(180)
Повернуться к	Повернуться в направлении другого существа или координат мышки. После команды всегда существует перечень объектов, которые в данный момент присутствуют в системе и на которые можно реагировать. В самом простом случае, когда других объектов нет, предлагается повернуться в сторону, где находится указатель мышки.
Изменить X на()	Изменить положение по оси X на указанное число шагов. Например: Изменить X (-10) – объект смещается влево на 10 шагов
Установить x в ()	Установить объект положение по оси x. Например: Установить x в (50)
Идти в x () y()	Переместиться в точку с указанными координатами Например: Идти в x(70) y(-50)
Плыть () сек в точку	Плавно переместиться в точку с указанными координатами за указанное время. На перемещение затратится время указанное в секундах. Например: Плыть(20) сек в точку x(90) y(90) Объект медленно (в течение 20 секунд) будет переползать в указанную точку.
Идти в ()	Перейти в точку, где расположен указатель мыши или другая фигура.
Если край, оттолкнуться	Если попадаешь в край экрана, то отразись от него. Это очень полезно, если Вы не хотите потерять своего героя
Следующие несколько блоков нельзя использовать в качестве отдельных строительных блоков, но они могут быть вставлены внутрь строительных кирпичиков в качестве входных параметров	
Положение x	Возвращает значение по оси X Например: Установить y (положение x)
Положение Y	Возвращает значение по оси y Например: Установить x (положение y)
Направление	Возвращает направление

Команды рисования (темно-зеленый ящик)

В зеленом ящике собраны команды, которые позволяют управлять пером или карандашом, которым могут рисовать наши объекты. Как и черепашка Лого, все объекты Скретч могут рисовать и оставлять на экране свои отпечатки.

Мы можем менять цвета и размер карандаша, поднимать и опускать карандаш. В результате выполнения сценария на экране рисуются разноцветные геометрические фигуры.

Команда	**Назначение**
Очистить	Очистить экран от всех следов, которые на нем оставили объекты
Опустить перо	Опустить карандаш. После этой команды за движущимся объектом будет оставаться след.
Поднять перо	Поднять карандаш
Установить цвет пера()	Выбрать цвет, которым мы собираемся рисовать. Цвет можно выбрать из объектов на экране
Изменить цвет пера на ()	Изменить цвет пера на указанное значение. Например: Изменить цвет пера на (50)
Изменить тень на ()	Установить градиент цвета следа, который оставляет объект. 100 – белый, 0 – черный.
Печатать	Отпечатать объект на экране

Кирпичики чисел (зеленый ящик)

Кирпичики чисел и датчики - все это бойцы невидимого фронта. А кирпичики чисел можно использовать только внутри строительных блоков. Сами по себе как строительный материал эти кирпичики использовать нельзя. Они возвращают результат. Как правило, все арифметические и логические блоки используются вместе с блоками управления.

Команда	**Назначение**
() + ()	операция сложения
() – ()	вычитание
() * ()	умножение
() / ()	деление
() > ()	Сравнение больше, равно, меньше. Например: ((5) * (5)) < ((4) * (10))
Выдать случайное от () до ()	Возвращает число в интервале от () и до (). Например: Установить направление(Выдать случайное от (-180) до (180)
Логические операторы. Эти блоки содержат входные окошки, куда вставляются кирпичики сравнений.	
<> и <> <> или <> Не <>	Например: () и ()
Функция от ()	Перечень возможных функций показан на рисунке после этой таблицы

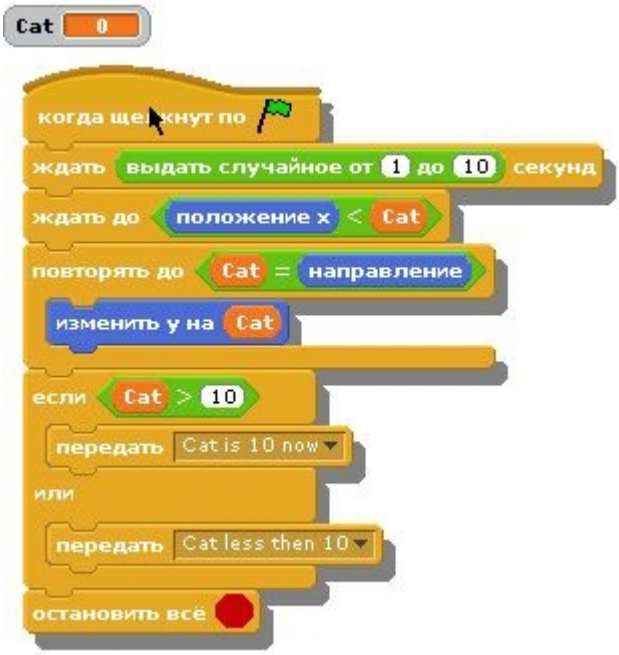
Возможные вычисляемые функции

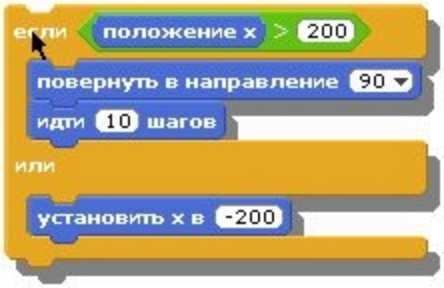
abs
квадратный корень
sin
cos
tan
asin
acos
atan
ln
log
e ^
10 ^

Посмотреть на результат выполняемой функции можно, если попросить котенка или другого героя сказать или подумать результат функции:

Кирпичики контроля (желтый ящик)

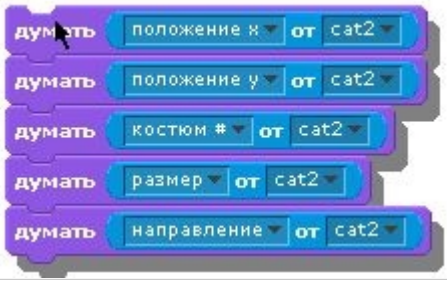
Кирпичики контроля позволяют собирать блоки команд и управлять последовательностью исполнения скриптов. Они определяют то, при каких условиях совершается то или иное действие. Они позволяют спрайтам передавать, принимать и реагировать на управляющие сообщения.



Команда	**Назначение**
Когда щелкнут по флажку	Запускает скрипт в ответ на щелчок по зеленому флажку. Это обычный способ запуска всей модели.
Когда нажата () клавиша	Запускает выполнение блока команд в ответ на нажатие выбранной клавиши. Позволяет передать управление на клавиатуру. Например: когда нажата стрелка вверх, повернуть вверх
Ждать ()	Команда ожидания. Параметр () указывает сколько секунд следует ждать.
Всегда	Блок команд, заключенных внутрь конструкции будет выполняться постоянно.
Если <>	Условие, при выполнении которого должны выполняться команды, заключенные внутри конструкции. Если не выполняется, то никаких действий
Если <> или	Условие, при выполнении которого должны выполняться команды, заключенные внутри конструкции **если**. Если не выполняется, то нужно перейти к действиям внутри блока или. 
Повторить()	Повторение. Параметр указывает, сколько раз нужно повторить блоки команд, заключенные внутри блока повторить() Например, повторить 20 (идти 50 повернуть 45)
Передать ()	Передать сообщение. Переданное сообщение может запускать активность другого исполнителя. Работает в сочетании с командой «Когда я получил ()»
Когда получил ()	Запускает выполнение блока команд в ответ на полученное сообщение. Работает в сочетании с «передать ()»
Всегда если ()	Выполняется до тех пор, пока условие () верно
Ждать до ()	Ждать, пока не выполнится условие
Остановить скрипт	Остановить выполнение программы для данного исполнителя
Остановить всё	Остановить выполнение всех программ

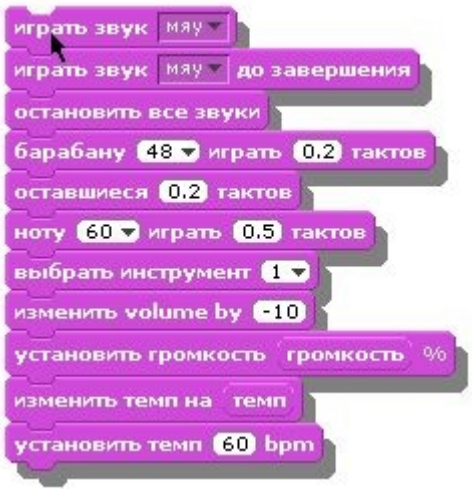
Датчики (голубой ящик)

Кирпичики из голубого ящика тоже нельзя использовать в качестве строительных блоков, но могут использоваться как и числа и логические значения «истина» или «ложь». Для того чтобы посмотреть на значение датчика можно попросить котенка проговорить или продумать значение датчика. Интересно, что датчик может получать информацию не только от объекта, с которым он связан, но и от любого объекта, существующего в нашей системе.

Команда	Назначение
Мышка по x	возвращает значение указателя мыши по оси x
Мышка по y	возвращает значение указателя мыши по оси x
Касается цвета ()	касается ли наш объект указанного цвета Например: touching color () ? –
Касается цвет () цвета	Касаются ли указанные цвета. Возвращает «истина» или «ложь»
Мышка нажата	Нажата ли управляющая клавиша мышки? Возвращает «истина» или «ложь»
Касается ()	Касается ли наш объект мышки или другого существа касается ли наш объект цвета() соприкасается ли цвет() с цветом().Возвращает «истина» или «ложь»
Параметр () от ()	Возвращается размер, положение по x или y, направление, костюм и громкость любого объекта существующего в данной модели.
Расстояние до ()	Возвращает расстояние от нашего спрайта до выбранного объекта – край, мышка, другой спрайт. 

Лиловый ящик - Звуки

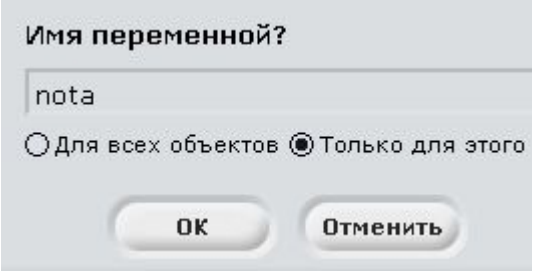
Лиловый ящик Скретч предоставляет массу звуковых возможностей. При этом звук можно выбрать в библиотеке – там этих звуков достаточно много. Так же как и библиотека картинок, библиотека звуков расширяется и к ней можно добавлять свои звуки в формате wav. Звуки можно записывать в редакторе самой среды Скретч. Внутрь звуковых кирпичиков можно добавить любые значения, которые мы вычисляем или считываем с датчиков. К разным объектам в Скретче можно добавлять свои музыкальные партии.



Команда	Назначение
Играть звук	Воспроизвести звук (можно выбрать звук). При этом звук никак не влияет другие действия нашего объекта
Играть звук до завершения	Играть звук до его завершения. Никакие другие действия наш спрайт при этом не совершает.
Выбрать инструмент	Выбрать инструмент, который будет играть. Инструментов в Скретче более 130
Барабанам	Ударные играют указанное число тактов
Играть ноту	Играть определенную ноту указанное количество времени (в секундах). Ноты записаны в цифрах, но против каждой цифры стоит ее звучание.
Убрать все звуки	Все звуки данной модели умолкают
Изменить темп	Меняет темп игры
Темп	Возвращает значение темпа игры

Переменные (оранжевый ящик)

В оранжевой ящике изначально ничего не лежит, кроме предложения «создать переменную». Мы сами создаем кирпичики, которые планируем использовать в своих сценариях. Если мы щелкаем по кнопке «создать переменную», то Скретч предлагает определить имя и свойства новой переменной. Переменные могут быть глобальными и локальными.



Глобальные переменные действую для всех объектов внутри данного проекта. Локальные переменные действуют только для данного объекта. Например, мы хотим несколько раз сыграть несколько нот, постепенно убыстряя темп игры. Мы создаем две локальные переменные – «nota» и «temp”

Temp – определяет время звучания и увеличивая значение этой переменной мы ускоряем скорость игры.

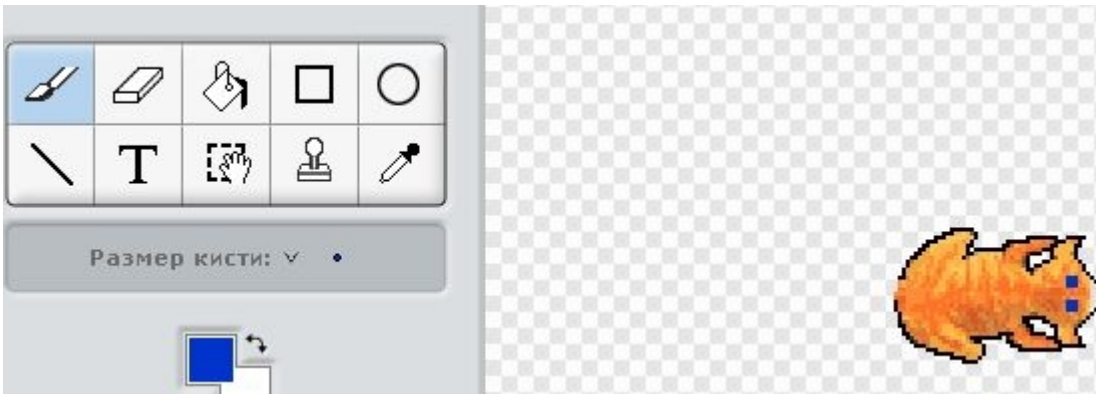
Примеры рецептов в среде Скретч

Рецепт № 1 Мультипликация

Что нам потребуется для мультипликации:

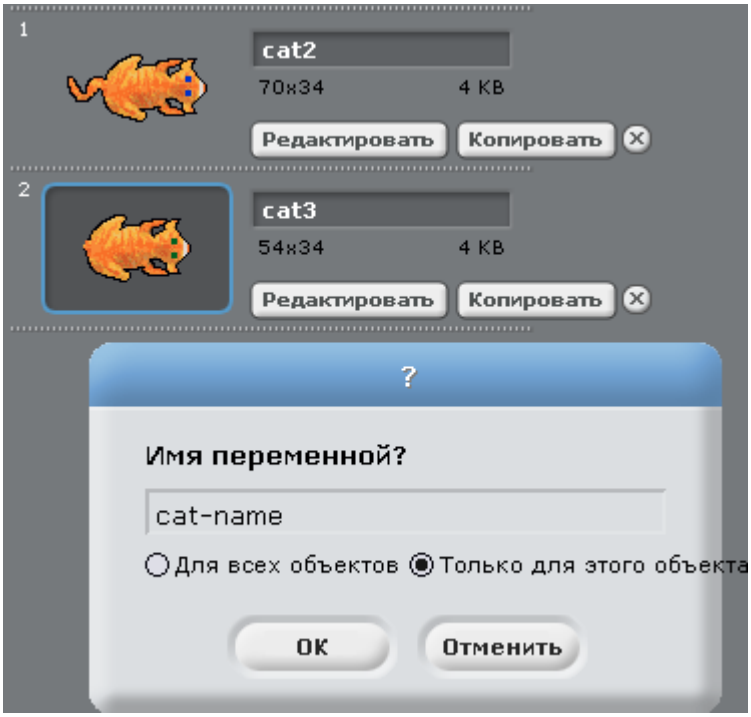
- Костюм кошки cat2 из библиотеки Animals
- Графический редактор Скретч
- Кирпичики из фиолетового ящика внешности
- Переменные, которые создадим в ящике «Переменные»

Мы берем костюм кошки из библиотеки, переходим в режим костюмы и копируем костюм cat2. У нас появляется костюм cat3 – точная копия cat2. Мы создадим две версии костюма кошки, и в одной версии кошка будет с голубыми линзами, а в другой с черными. Чтобы было заметнее, мы у голубоглазой кошки еще и хвостик оторвем. Эту копию мы редактируем в графическом редакторе Скретч. Кисточкой подрисовываем кошке синие глазки, ластиком отрываем хвостик.



Теперь попробуем управлять внешним видом кошки и сделаем это поведение неожиданным для себя. Для этого воспользуемся оранжевым ящиком переменных.

Ящик с переменными изначально пуст. Значения в этом ящике появляются только в результате наших действий. Мы создадим новую переменную с именем **cat-name**.



Обратите внимание, что при создании переменной программа спрашивает нас, хотим ли мы, чтобы эта переменная была глобальной – доступной для всех объектов, или эта переменная будет действовать только для данного объекта. В этом случае другие объекты ничего об этой переменной знать не будут.

После того как переменная создана, мы можем со значением этой переменной совершать действия. Мы можем присваивать переменной любые числовые значения и менять ее значения на числовые значения (увеличивать или уменьшать)

Присвоим переменной значение, которое выдаст датчик случайных чисел от одного до двух. Иногда этот датчик будет выдавать число 1, иногда 2.

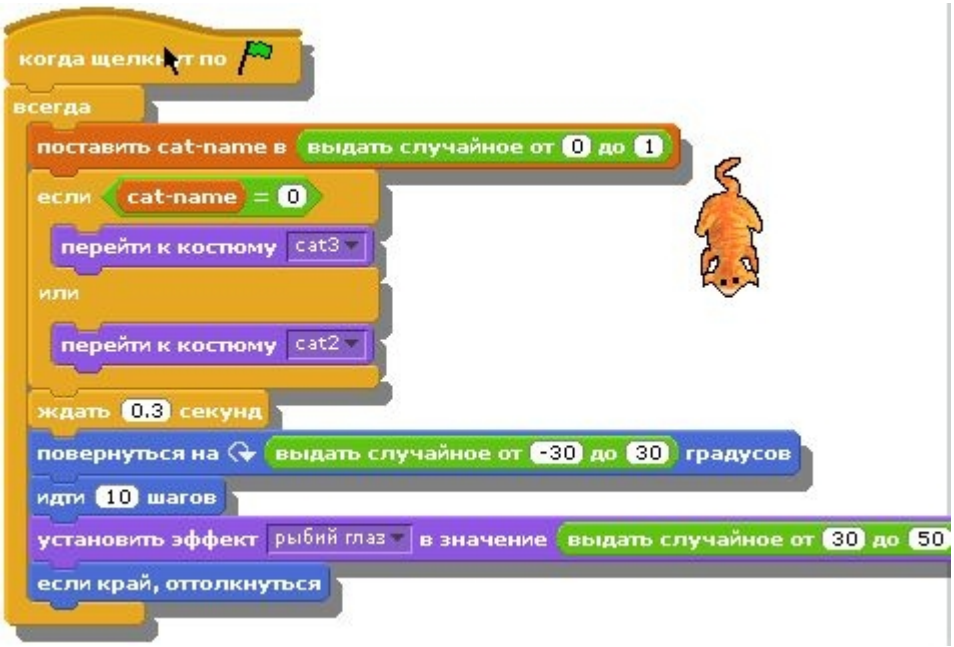
На нашем рисунке кошка в ответ на запуск программы говорит «0», но в среде Скретч она с равной вероятность говорит «0» и «1».Дальше мы поставим управляющие кирпичики блока из желтого ящика. Главное в этом рецепте – блок выбора «**Если или** »

- Если значение переменной = 1, то кошка надевает голубые линзы и прячет свой хвост.
- Если значение переменной = 2, то кошка надевает черные линзы

Теперь у нас на экране кошка мигает линзами разного цвета и поджимает хвостик.

Как может быть улучшен рецепт?

- Можно добавить из фиолетового ящика графический эффектыи кошка будет становить толще или искривляться.
- Можно добавить из синего ящика команду движения и кошка будет не только менять внешний вид, но и ходить по экрану.



Мы используем готовую кошку в следующем рецепте, где ей предстоит гоняться за мышкой.

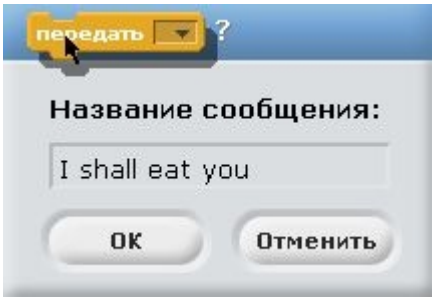
Рецепт № 2 Взаимодействие объектов – кошка убегает от мышки

Мы хотим собрать мультфильм, в котором кошка будет преследовать мышку.

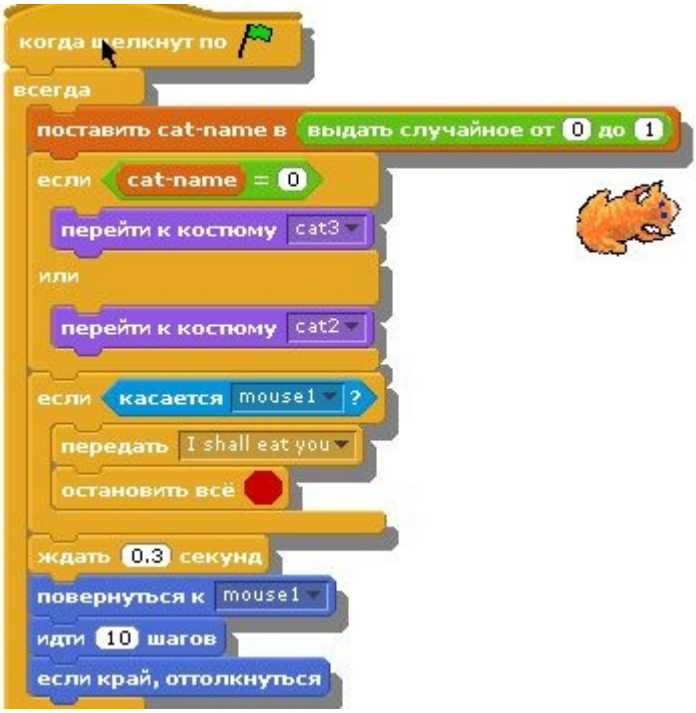
Что нам потребуется для мультипликации:

- Кошка, которую мы уже сделали в предыдущем рецепте
- Мышка из библиотеки Animals
- Кирпичики обмена сообщениями из рыжего желтого ящика
- Датчики из голубого ящика

Действия кошки из предыдущего рецепта мы несколько видоизменим. Во-первых, она будет перемещаться не случайно, а в направлении мышки. Во-вторых, мы добавим условие завершения работы всей модели. Если кошка коснулась мышки, то она передает ей сообщения «Я тебя съем» и заканчивает выполнение всех программ. Новым в этом рецепте будет создание нового сообщения:



По созданному сценарию кошка поворачивается в направлении к мышке, двигается к ней и успешно ее ловит.



Теперь мы хотим усложнить кошке задачу и научить мышку убегать от кошки.

Во-первых, пусть кошка в начале игры появляется в случайном месте экрана. Для этого мы используем команды **установить x** и **установить y** и передаем им случайные значения внутри рабочего экрана.

Во-вторых мы научим кошку исчезать, если кошка ее поймала:

Когда я получу сообщение «Я тебя съем» - спрятаться.

В третьих, мы научим мышку убегать от кошки

Для этого мы используем для датчика. Мышка будет спокойно сидеть на месте, пока кошка не подойдет к ней на расстояние 90 шагов. Как только это случиться, мышка повернется в ту же сторону, что и кошка, несколько отклонится от этого направления и перебежит в новую точку.

В результате мы получили свою первую игровую модель, с которой можно экспериментировать.

Как может быть улучшен рецепт?

- Мы можем подобрать значения угла поворота, при котором мышка никогда не попадется в лапы кошке.
- Мы можем поместить мышку и кошку в среду, где им придется решать дополнительные задачи – обходить препятствия, реагировать на цвета и т.д.

Рецепт № 3 Играем с геометрическими фигурами

Кошка Скретч является потомком древней черепашки Лого и будет вежливо по отношению к старшим родственникам показать, что и наша кошка может рисовать геометрические фигуры не хуже черепашки. Большинство геометрических проектов, которые выполнялись с черепашками Лого и другими исполнителями, могут быть успешно реализованы и в среде Скретч.

Что нам потребуется для такого рисования:

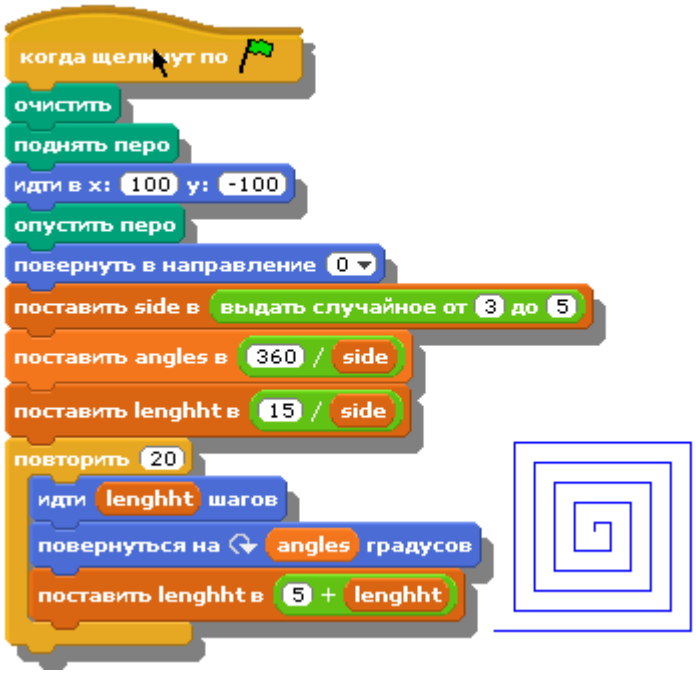
- Управляющие блоки – «всегда» и «повторить» из желтого ящика
- Переменные, которые мы сами создадим в оранжевом ящике. В этом проекте 2 локальные переменные: число сторон многоугольника. И угол поворота. Число сторон мы задаем сами, а угол поворота подстраивается под наш выбор
- Кирпичики перемещения из синего ящика
- Кирпичики рисования из зеленого ящика

В этом проекте мы учим нашу кошку рисовать фигуры:

- выбираем случайное число сторон фигуры N (3, 4, 5)

- в зависимости от числа сторон определяем величину угла (360 / N)

- рисуем фигуру, постоянно увеличивая длину ее стороны.



В начале работы мы очищаем экран и устанавливаем нашего исполнителя в определенную точку на экране. Эта последовательность:

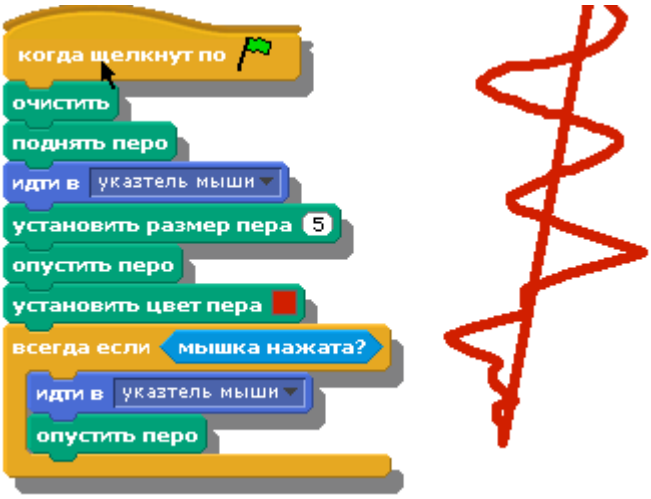
- Очистить
- Поднять перо
- Идти в () ()

Будет использоваться в проектах достаточно часто. Затем позволяем исполнителю выбрать из интервала заданных чисел сколько сторон будет у исходного многоугольника. Длина стороны подстраивается под количество сторон. На каждом шаге после поворота исполнитель увеличивает размер стороны на 5 единиц. В результате на экране появляется красивый узор.

Как может быть улучшен рецепт?

- Мы можем изменять не только длину шага, но и размер пера, которым рисует исполнитель и градиент цвета.
- Мы можем добавить к этому рецепту кирпичики из голубого ящика и начать управлять поведением исполнителя, настроив его на нажатие мышки.

На следующем рисунке показано, как мы можем рисовать на экране.



Мы проиграли в красоте линий, которые наш исполнитель оставляет на экране, но выиграли в интерактивности. Этот рецепт может улучшаться в сторону создания собственного графического редактора.

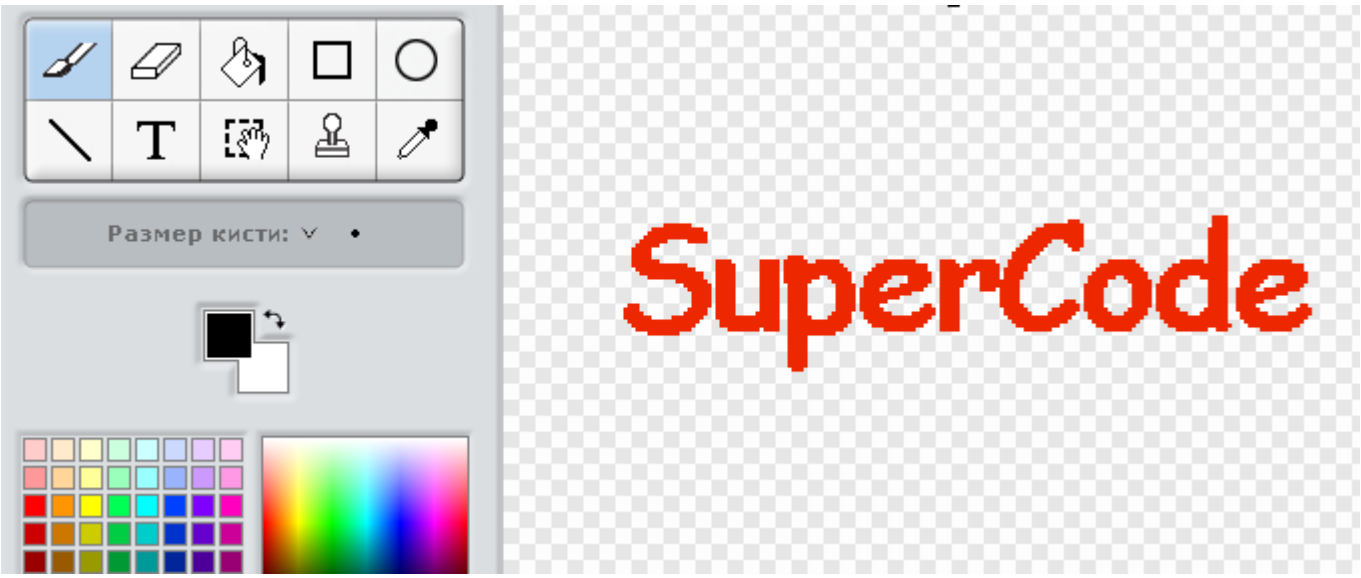
Рецепт № 4. Играем с буквами

В этом проекте мы продолжим рисовать на экране, но будем делать это при помощи букв. Мы хотим написать на экране слово для конкурса Суперкод:

Что нам потребуется для этого рецепта:

- Костюмы из галереи буквы – Letters
- Графический редактор Скретч
- Команды контроля
- Кирпичики рисования
- Кирпичики движения

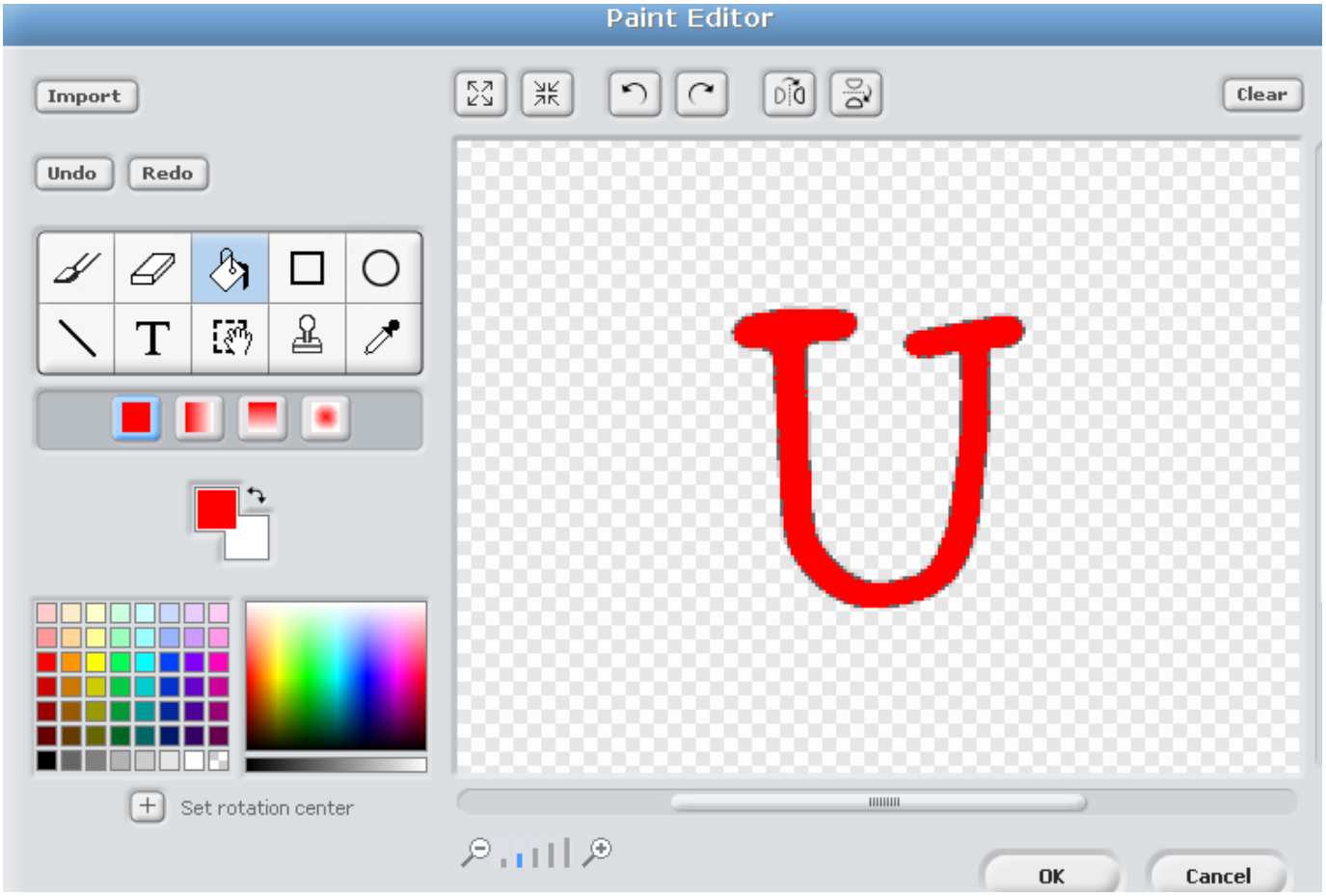
К сожалению Скретч пока не позволяет нам писать на экране русскими буквами. Самый простой способ написать текст на экране это – создать спрайт, на костюме которого будут написано нужное слово. Например, мы можем набрать нужное слово в графическом редакторе.



Но мы легким и одноразовым путем не пойдем. Мы соберем в костюмы несколько нужных букв. Мы создали новый объект и загрузили в него несколько форм, в которые он может наряжаться:

S U P E R C O D

Новые формы или маски добавляются кнопкой Импорт. Мы можем редактировать формы - кнопка Редактировать. В данном случае мы просто поменяли цвет букв с черного на красный.



Написать слово

Теперь мы хотим написать слово SuperCode

- Встань в центр экрана (x=0, y=0)
- Надень на себя маску **S**
- Окрась свой карандаш в белый цвет (чтобы не оставлять следов на экране)
- Повтори 8 раз
- Опустить карандаш
- Отпечатать свое изображение
- Поднять карандаш
- Надень маску следующей буквы
- Пройти вперед 18 шагов

Для последней буквы **E** делаем специальные команды - надень маску + отпечатай свою форму на экране

Написать слово по кругу

Пришла пора использовать возможности программирования. Уж если мы что сделали что-то один раз и собрали готовый блок, то можем повторить несколько раз:

Все, что было сделано для одного слова, повторим 10 раз. После каждого слова будем поворачиваться на 36 градусов.

Теперь мы умеем обращаться с буквами на экране и можем создавать из них геометрические фигуры.

Как может быть улучшен рецепт?

Рецепт может развиваться, если мы пополним библиотеку русскими буквами и будем писать на русском языке. Еще одно направление развития этого рецепта представлено в следующем рецепте «буквописателя».

Рецепт № 5. Поиграем со случайными надписями на экране

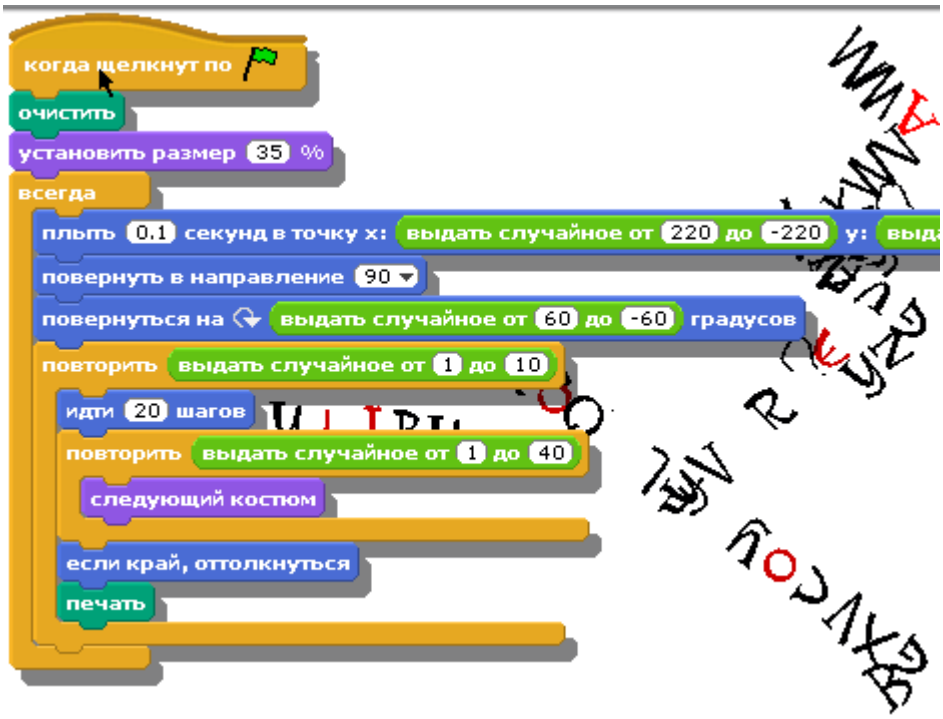
Мы продолжим рисовать буквами на экране и создадим безумного буквописателя, который будет заполнять стену буквами.

Что нам потребуется для создания такого писателя**?**

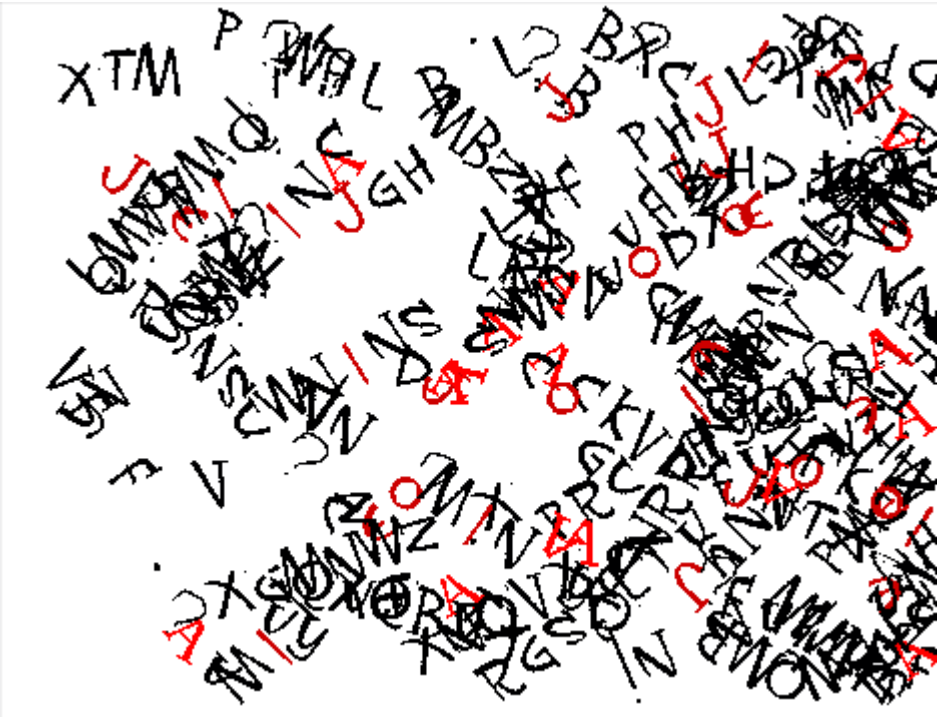
- Две уже знакомые команды из зеленого ящика: очистить и печатать.
- Команды из фиолетового ящика: установить размер и следующий костюм.
- Команды из оранжевого ящика: всегда и повторить.
- Случайное значение из ящика чисел.

Главная хитрость рецепта состоит в получении случайной буквы, которая будет отпечатываться на стене. Для получения этой буквы мы пользуемся конструкцией:

Поменяй костюм случайное число раз от 1 до 40



В результате наш писатель успешно справляется с поставленной задачей и бодро разукрашивает стенку загадочными словами:

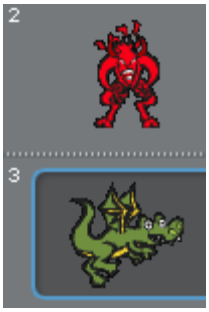


Как может быть улучшен рецепт?

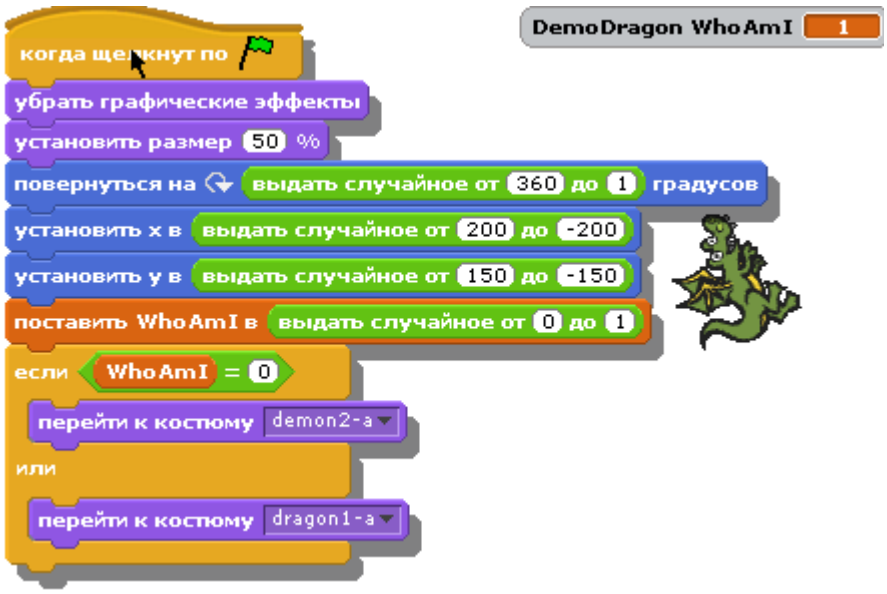
Развитие проекта может идти в сторону написания более осмысленных фраз.

Рецепт № 6 Модель битвы драконов и демонов

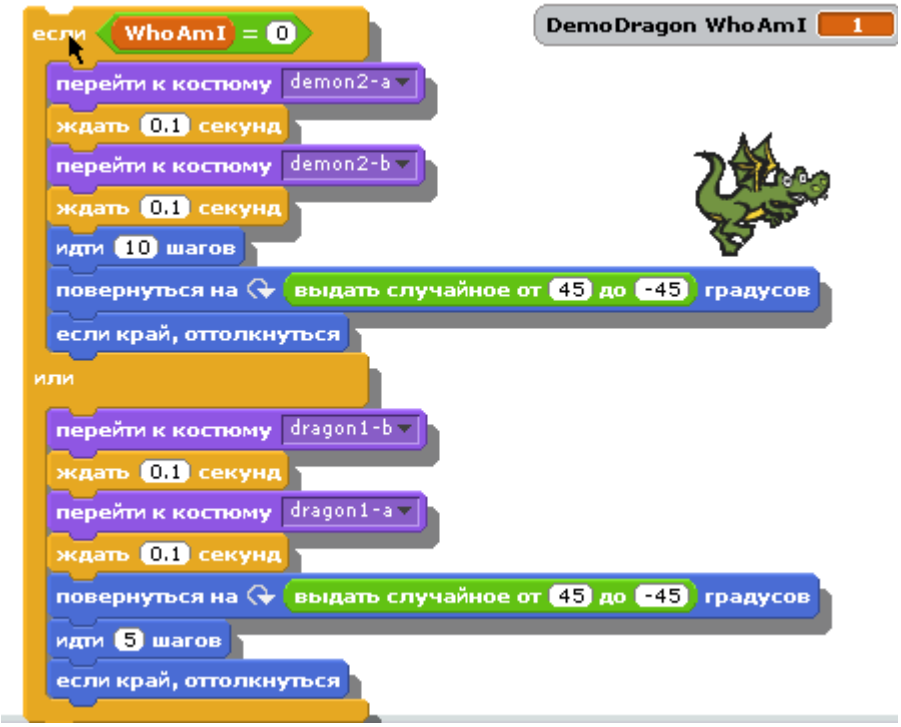
От простого проекта, в котором кошка меняла цвет линз мы перейдем к проекту со множеством действующих персонажей. Мы еще в самом начале говорили, что Скретч это мультиагентная среда, но ни разу еще не показали этого. Теперь время пришло. Нам понадобятся знакомые конструкции, и в начале мы будем иметь дело только с одним объектом. Этот объект может надевать на себя два разных костюма – дракона и демона:



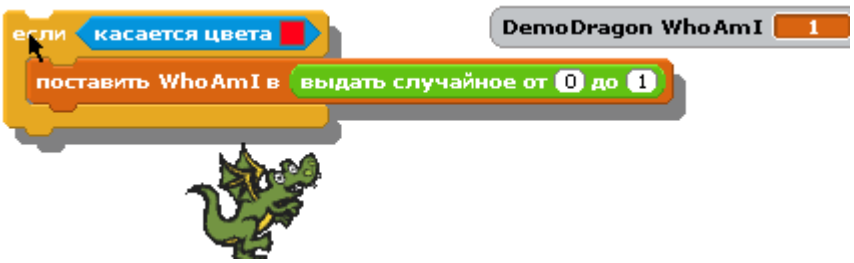
По щелчку зеленого флажка наш герой занимает случайное положение на экране и надевает на себя одну из масок – дракона или демона. Маска зависит от значения переменной (как это уже было с кошкой).



Теперь мы разрешим нашему герою перемещаться по экрану: герой меняет костюмы, проходит 10 шагов, поворачивается в случайном направлении и не покидает пределов экрана. Если он родился в костюме демона, то все эти действия он и совершает демоном. Если он родился драконом, то он ведет себя как дракон.



Теперь мы введем неожиданное предположение, что наш герой может встретить игрока другой команды. Например, зеленому дракону на пути может встретиться красный демон и он под воздействием этой встречи с красным может превратиться в демона.



Или наоборот, красному демону может встретиться зеленый дракон и он может под воздействием встречи с зеленым может превратиться в дракона:

Подчеркиваем, что превращение носит необязательный характер. Может превратиться или не превратиться с вероятностью 50%.

Ниже представлен фрагмент действий нашего героя, если он действует в обличии демона:

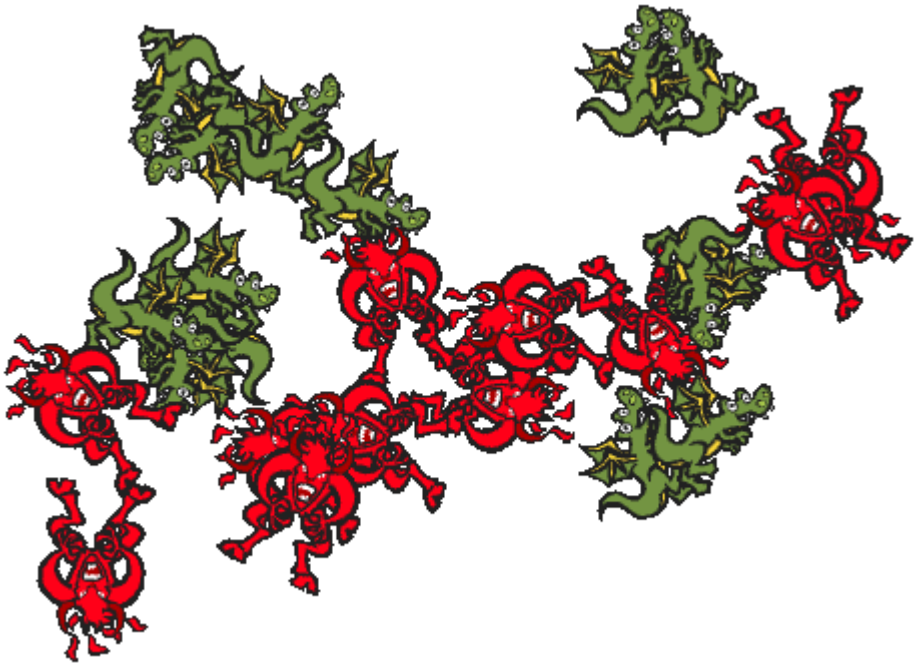
Команда на этом не заканчивается и ниже идет запись ИЛИ:

Вести себя как дракон, а если коснешься красного демона, то можешь превратиться в демона.

И теперь возникает справедливый вопрос: откуда возьмутся все эти другие демоны и драконы, которых наш герой возможно встретит? А мы их сейчас откопируем из нашего же героя. У нас есть специальная кнопка экспорта отдельного спрайта и мы потом можем

взять готовый спрайт в качестве нового героя. Мы можем таких спрайтов взять множество, и все они будут между собой взаимодействовать.

На следующем рисунке представлено поле боя, на котором бьются 25 драконов и демонов. Все они – копии этого маленького спрайта.



Это достаточно сложный пример, в котором показаны возможности копирования и взаимодействия множества объектов внутри среды Скретч. Дальнейшее развитие модели может идти в сторону уточнения поведения драконов и демонов. Может быть, они будут переходить из одной команды в другую только встретив нескольких противников.

Рецепт № 7 Управление через обмен сообщениями

В простой среде Скретч видны мощные возможности параллельных действий множества исполнителей. В этом проекте взаимодействует несколько героев:

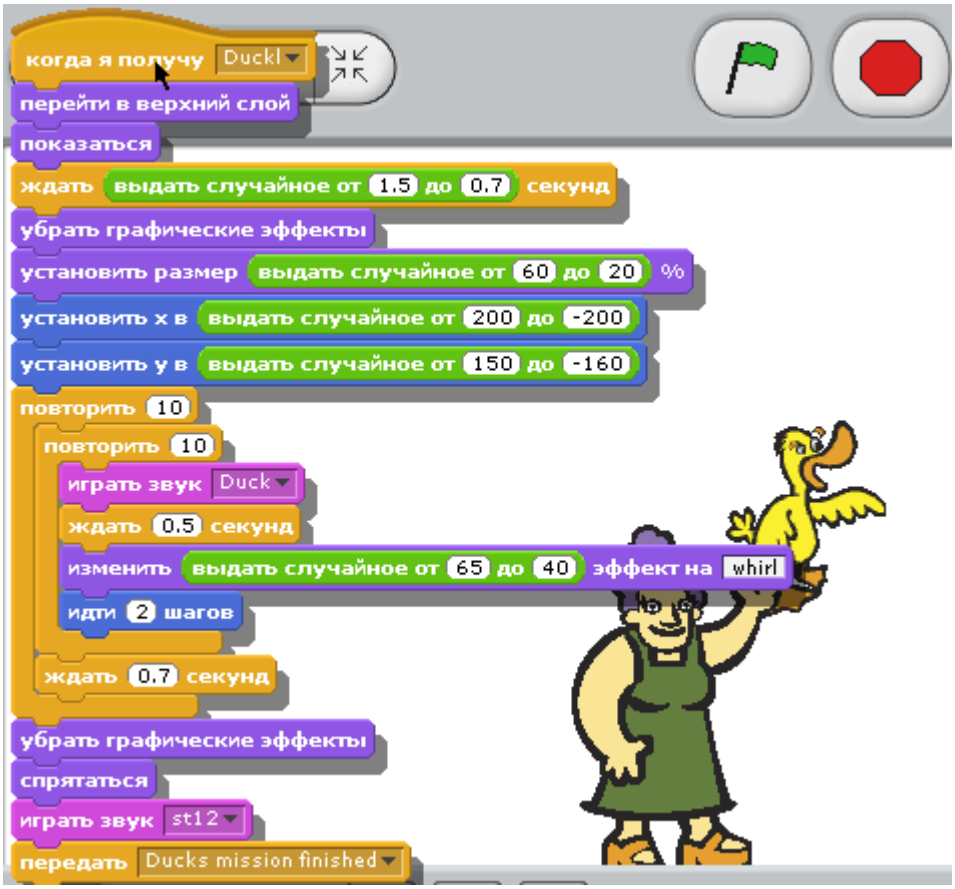
- Приходит женщина, поет свою песенку
- После женщины приходят утки, выполняют свою работу.
- После уток приходят коровы, выполняют свою работу.
- После коров приходят собаки, выполняют свою работу.
- После собак приходят лягушки, выполняют свою работу.
- После лягушек приходят лошади, выполняют свою работу.

На следующем рисунке приведен фрагмент экрана программы, на котором видны массы исполнителей.

Основная трудность управления заключается в синхронизации действий. Новые исполнители не должны приходить, пока предыдущая группа не выполнит свою задачу. В Скретче проблема взаимодействия решается через обмен сообщениями. Все исполнители взаимодействуют между собой при помощи двух "коммуникационных" блоков:

- **Передать** - команда передать определенное сообщение. Например, по завершении своих действий группа лошадей может передать сообщение "Лошади выполнили свою задачу"
- **Когда я получу** - блок, который ожидает определенное сообщение. Например, если блок получает сообщение " Лошади выполнили свою задачу", то он может запустить следующих героев мультипликационной истории.

.



Проект простой, но механизм организации взаимодействия групп агентов дан в нем достаточно наглядно.

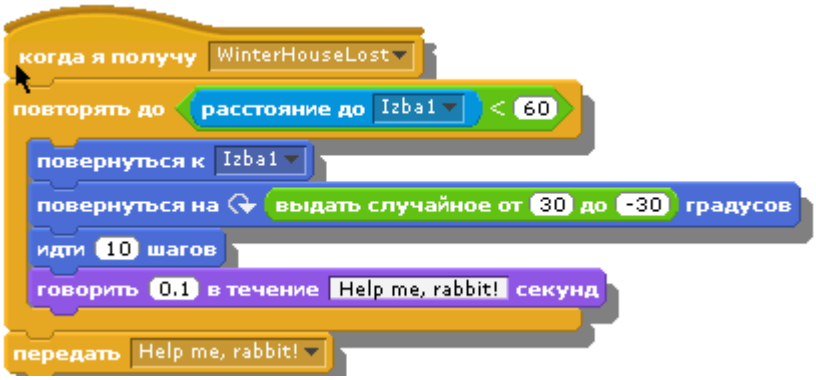
Рецепт № 8 Сказка про Зайца и Лису

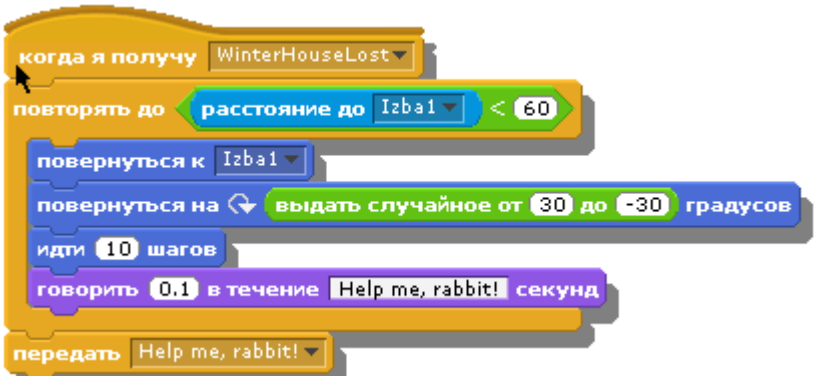
Этот проект продолжает идею обмена сообщениями между различными агентами и в нем представлен подход, когда мы заранее описываем в таблице множество всех действующих агентов. Интересен как рецепт предварительной записи действий всех героев.

Исполнитель	Действия
Лубяная избушка	Стоит (координаты) и ни на что не реагирует
Ледяная избушка	Стоит (координаты). Получив сообщение от Весны, тает. Сообщает о собственном таянии
Весна	Ждет. Приходит. Сообщает о своем приходе
Заяц	Живет в лубяной избушке Получает сообщение - Помоги лисе Посылает сообщение - Лиса, заходи Получив сообщение - Уходи из избушки, покидает избушку, начинает плакать, посылает сообщение - "Я плачу в первый раз" Получив сообщение - О чем, заяц, плачешь? Посылает сообщение - "Как же мне не плакать?" Получив сообщение - "Собака убежала", посылает сообщение "Я плачу во второй раз" Получив сообщение - "Кошка убежала", посылает сообщение "Я плачу в третий раз" Получив сообщение - "Лиса ушла", перестает плакать
Лиса	Живет в ледяной избушке Получив сообщение - "Я растаяла", идет к лубяной избушке, посылает сообщение "Помоги лисе" Получив сообщение - "Лиса, заходи", заходит внутрь лубяной избушки, посылает сообщение "Уходи из избушки" Получив сообщение "Уходи лиса вон", посылает сообщение "Как выскочу" Получив сообщение "Уходи лиса с печи", покидает избушку, посылает сообщение "Лиса ушла"
Собака	Получив сообщение "Я плачу в первый раз" , идет к зайцу посылает сообщение "О чем, заяц, плачешь?"

	Получив сообщение "Как же мне не плакать?" , говорит "Не плачь, заяц" идет к избушке, посылает сообщение "Уходи лиса вон" Получив сообщение "Как выскочу", убегает, посылает сообщение "Собака убежала"
Кот	Получив сообщение "Я плачу во второй раз" , идет к зайцу, посылает сообщение "О чем, заяц, плачешь?" Получив сообщение "Как же мне не плакать?", говорит "Не плачь, заяц", идет к избушке, посылает сообщение "Уходи лиса вон" Получив сообщение "Как выскочу", убегает, посылает сообщение "Кошка убежала"
Петух	Получив сообщение "Я плачу в третий раз" идет к зайцу посылает сообщение "О чем, заяц, плачешь?" Получив сообщение "Как же мне не плакать?" говорит "Не плачь, заяц", идет к избушке, посылает сообщение "Уходи лиса с печи".

Из таблиц действия легко превратить в команды Скретча.

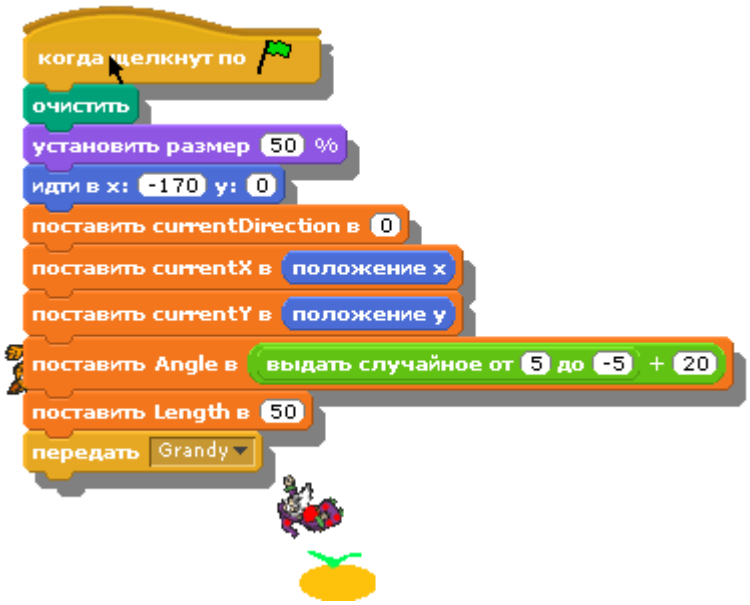




Рецепт № 9Сказка про репку

В этом проекте мы объединяем обмен сообщениями между группами агентов и сохранение текущих положений и направлений в переменных.

От классического варианта сказки наш проект отличается еще тем, что репка по-прежнему одна, но вытаскивать ее пришли 2 дедки. Каждый из дедок позвал на выручку по 2 бабки, каждая бабка позвала по паре Жучек, каждая Жучка прихватила пару кошек и каждая кошка прихватила по паре летучих мышек. Внучек в программе нет вообще. Выполнение программы начинается с репки, которая занимает положение на экране, устанавливает значения и посылает сообщение дедке.



Дедка получает сообщение, идет в указанную точку, оставляет там свой след, посылает бабке сообщение и ждет от нее сигнала о завершения операции. Здесь очень важная конструкция: **Передать «Бабка» и ждать**

По завершению операции

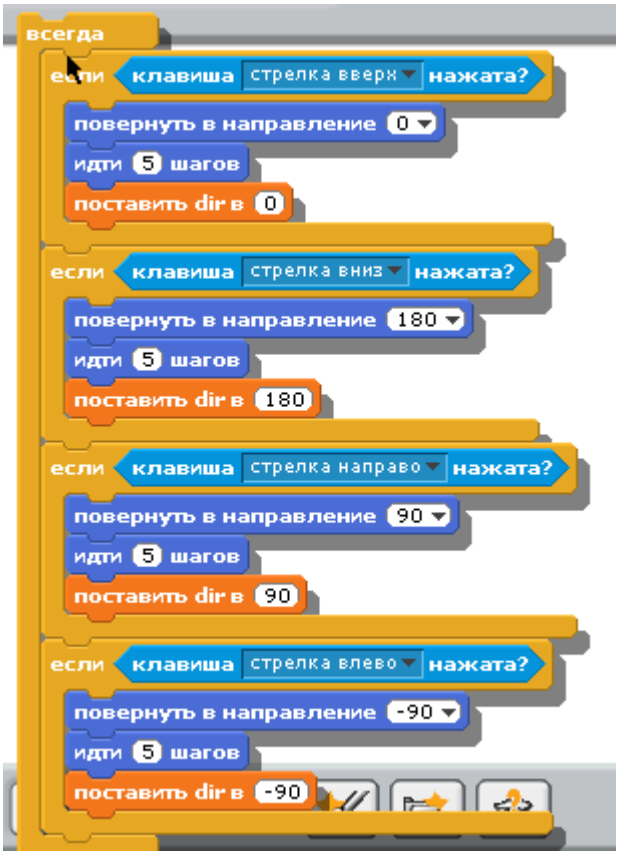
Каждый следующий агент повторяет и умножает действия предыдущего.

Рецепт № 10 Скретч-квест

Квест, приключенческая игра — один из основных жанров компьютерных игр, требующих от игрока решения умственных задач для продвижения по сюжету. Сюжет может быть предопределённым или же давать множество исходов, выбор которых зависит от действий игрока.

Квест предполагает, что герой перемещается из одного пространство в другое. Число пространств или комнат может быть очень большим, но для примера мы построим мир из 5 связанных между собой комнат. Кошка управляется клавишами - вниз, вверх, вправо и влево.

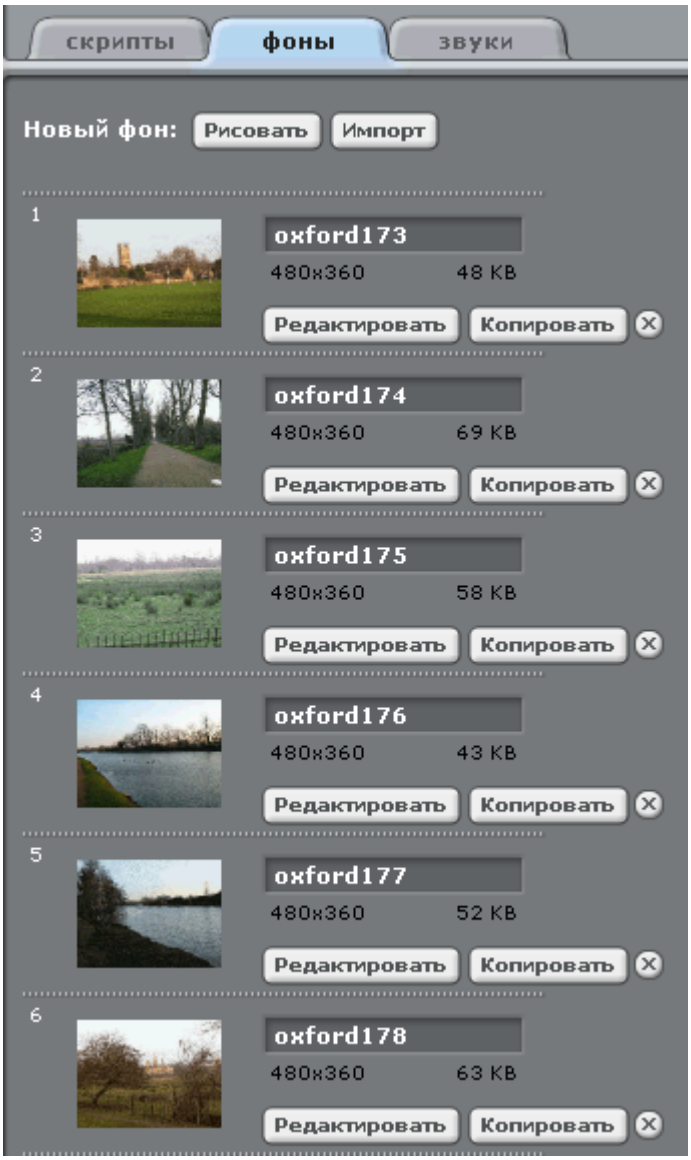
Если клавиша стрелка вверх, то повернуться вверх пройти 5 шагов и установить значение переменной 0. Все остальные нажатия – вправо, влево и вниз – отрабатываются точно так же.



С кошкой все ясно. А вот мир вокруг кошки устроен сложнее, он должен помнить, откуда идет кошка и куда она должна попадать при ее соприкосновении с экраном.

Как только кошка касается экрана, она посылает миру сигнал – **передать "Change place"**

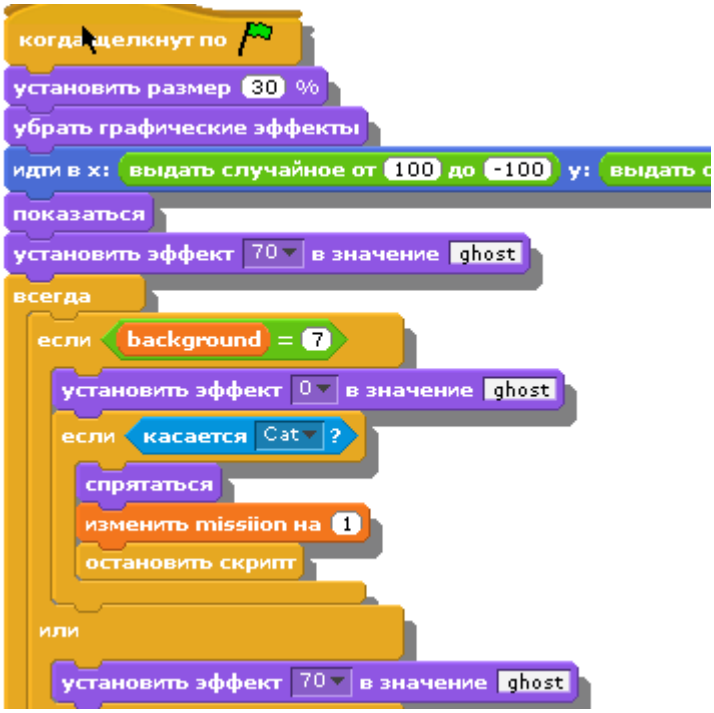
В ответ на этот сигнал среда начинает анализировать положение (направление) кошки и фон, на котором она находится. Все изображения – костюмы фона.



В зависимости от того, куда смотрит кошка, и из какой комнаты она пришла, фон подсовывает под кошку новое изображение окружающего мира.

Теперь мы можем путешествовать из комнаты в комнату или из области в область. Усложним игру, добавив к кошке двух злобных демонов. Каждый из них постоянно поворачивается в сторону кошки и двигается к ней. Если один из демонов касается другого, он замирает. Если демон коснется кошки, то мы проиграли.

Добавим кошке миссию, которую она должна выполнить, переходя из одной области в другую: найти и собрать семь зеленых шариков. Шарики должны быть видимы в некоторых областях и невидимы в других.



Теперь у нас практически полноценный квест, в котором игрок бродит по полям, собирает зеленые шарики и ускользает от противников.

Проект можно расширять новыми заданиями и новыми областями.

Сообщество Скретч

В сети существует сообщество пользователей, которые обмениваются моделями, написанными на языке Скретч.

Адрес сообщества - http://scratch.mit.edu/

Скретч интересен и сам по себе и тем сообществом, которое сложилось вокруг этого средства. В этом сообществе участники обмениваются и обсуждают результаты своей деятельности - конкретные объекты, программы, рисунки и т.п. В основании использования таких сообществ в образовании лежит философия конструкционизма и технология обмена идеями. Мы можем наблюдать, использовать и копировать небольшие фрагменты действий, которые ежедневно совершают в сети другие люди. В результате повторения простых операций мы можем через сеть постепенно перенимать способы деятельности людей умеющих и знающих больше чем нашего.

Мы можем находить в сети необходимые образцы поведения, о важности которых писал Питер Сенж «*На самом деле мы все рождаемся системными мыслителями, способными воспринимать и понимать системы, с которыми мы взаимодействуем. Редкий ребенок в четыре года не понимает, что если он не приберется в своей комнате, то это сделают мама или папа. Дети в этом возрасте очень хорошо понимают паттерн взаимосвязи поведения нескольких людей, которые поддерживают чистоту в комнате, точно так же как они понимают множество других социальных систем, в которые они включены в своих семьях и в сообществах своих друзей. Они учатся, делая, а не слушая. Они узнают о том, кем являются они сами. Вместо того чтобы протестовать против взрослых, которые требуют их внимания и послушания, они сами ищут взрослых, которые могут дать им нужный совет и служить образцами для поведения. И дети находят эти образцы повсюду».* (Senge P., 2000)

Сетевые сообщества, в которых общение происходит на основе и по поводу создаваемых участниками сообщества артефактов. В статье «Думать как дерево» Мич Резник подробно описывал деятельность такого сообщества:

Примеры проектов создаются самими членами сообщества. Поскольку каждый созданный объект можно изучить и скопировать, каждый такой объект является образцом для всех остальных. Если ученица видит интересный объект, она всегда имеет возможность посмотреть код программы, который управляет поведением данного объекта. Она может создать новую версию поведения объекта, модифицировав код программы-образца. Коллекция проектов образцов внутри учебного сообщества постоянно видоизменяется. Эти изменения следуют интересам учеников вне всякого централизованного контроля. Видоизменения коллекции происходят так же, как это происходит в экологических системах, - коллекция адаптируется к смене интересов участников сообщества. Если члены сообщества выказывают повышенный интерес к проектам определенного типа, то и коллекция примеров приспосабливается к этим изменениям.

Например, несколько членов сообщества заинтересовались волшебством. Один ученик создал волшебную палочку, другой ученик создал класс волшебных объектов, третий ученик создал книгу заклятий. Эта книга заклятий была наполнена простыми программами, которые могли накладывать различные заклятия на всех людей, находящихся внутри данной виртуальной комнаты. Многие из членов сообщества сделали копии книги заклятий, и многие добавили к ней свои собственные программы заклятия. Кто-то даже создал заклятие, которое запускало в действие все заклинания, существующие внутри всего сообщества. Эта популярная программа была скопирована многими учениками. Один ребенок открыл магазин, в котором можно было купить самые новые версии магических объектов. Постепенно интерес к магии был утерян. Число магических объектов внутри сообщества постепенно уменьшилось, автоматически приспосабливаясь к интересам сообщества.

Сообщество носит открытый характер. Любой желающий может посмотреть все материалы. Регистрация необходима только, если нужно загрузить на сервер свой готовый проект.

Разные люди начинают использовать Скретч по-своему. Некоторым нравится объединять блоки и смотреть, что получается. Некоторые предпочитают начинать с готовых проектов, которые входят в пакет Скретча, и вносят изменения в код этих проектов.

Представьте, что каждый проект это - пирог. Такой пирожок со своим собственным рецептом, (программными блоками - маленькими кирпичиками). Все члены сообщества

делятся своими пирожками и их рецептами. Это значит, что Вы можете попробовать пирожок и узнать, как испечь пирожки самому.

Нет никаких секретных рецептов, все рецепты открыты для всех. Каждый рецепт можно использовать и видоизменять. Рецепты могут служить источников вдохновения для новых идей. Вы можете есть пирожки и копировать чужие рецепты для создания своих собственных пирожков с другими составляющими. Эта свобода использования рецептов связана только двумя очень простыми требованиями или пожеланиями:

1. делитесь своими пирожками вместе с рецептами
2. отдавайте должное тем, кто вас вдохновил

Такой тип обмена внутри сообщества получил название Некоммерческая лицензия с указанием авторства — Копилефт. Пирожок сложнее кирпичика, он состоит из множества деталей и взаимодействие деталей может обеспечивать пирожку определенное поведение. Сообщество Скретч не идет дальше этих правил. Сообщество следует этим правилам, поскольку правила имеют смысл. Мы разрабатываем новые свойства сайта, которые помогают людям и делиться своими пирожками, и отдавать должное другим за их вклад. Например, Вы можете заметить, что всегда необходимо скачивать авторские блоки каждого проекта. Пожалуйста, всегда упоминайте людей, чьи проекты использованы в ваших собственных произведениях.

Что можно делать в сообществе Скретч:

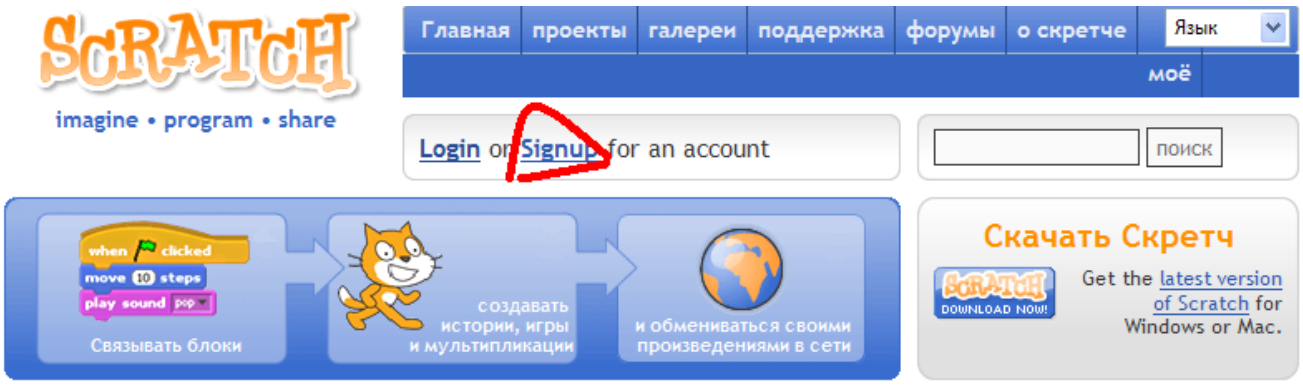
- Просматривать готовые проекты прямо в сети.
- Искать проекты - используются теги и галереи проектов
- Комментировать проекты
- Обмениваться сообщениями внутри сообщества Скретч
- Загружать проекты с сервера и запускать их на своем компьютере
- Видоизменять проекты
- Использовать фрагменты проектов (объекты), внутри новых проектов.

Иногда встречается формулировка - данная модель является продолжением или развитием модели - далее следует ссылка на предшествующую модель и имя её автора.

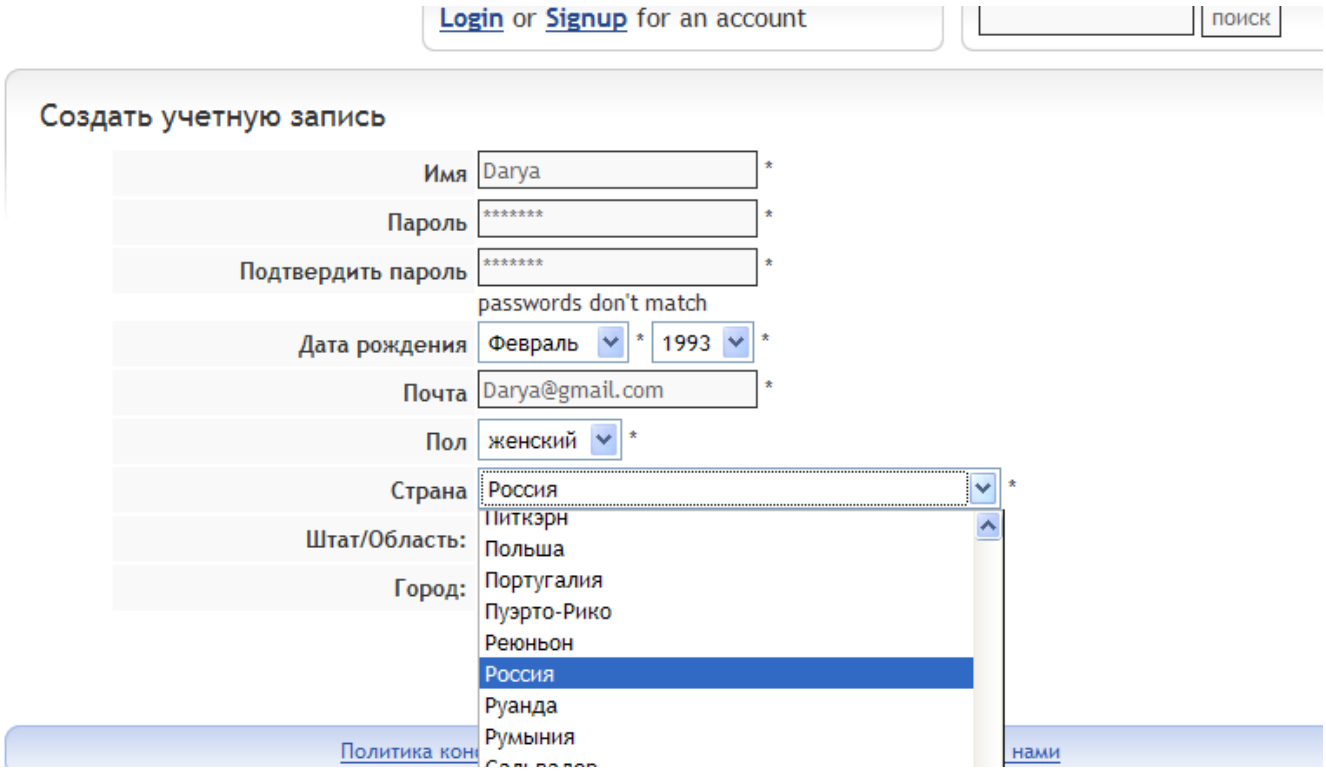
Изучение поведения и построение моделей более не ограничивается отдельным компьютером. Мы можем предложить свое произведение другим членам сообщества, и они смогут не только посмотреть, как работает модель, но и разобраться в том, как сложены кирпичики модели, смогут взять эти кирпичики и строительные блоки и построить из них свое собственное здание. На сайте http://scratch.mit.edu/ можно не только посмотреть готовые работы, но и скачать их код. После этого любой желающий может

видоизменить связь блоков программы и построить свою собственную, опираясь на плечи своих коллег

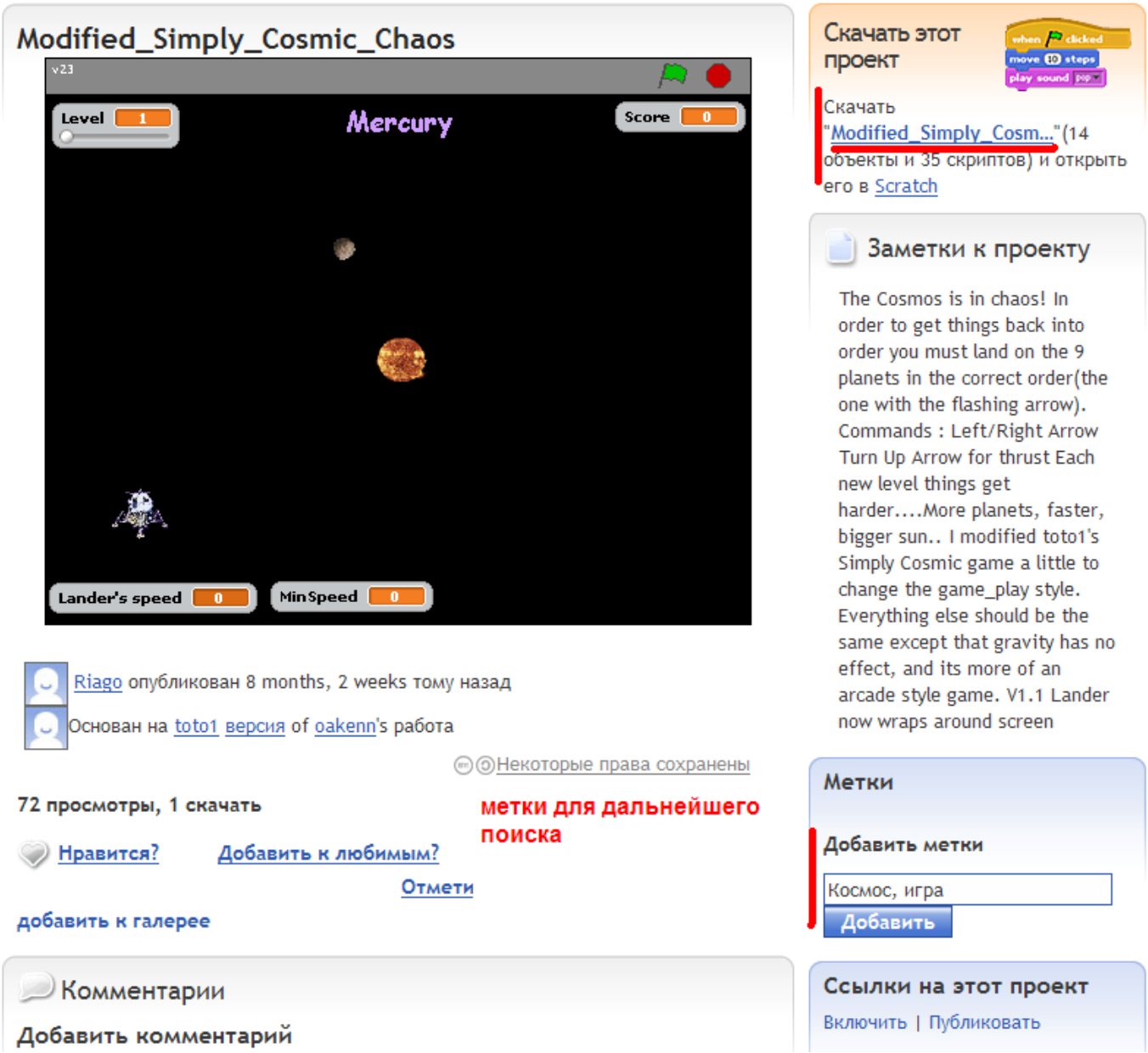
Как вступить в сообщество Скретч?



Далее все прозрачно - заполняете поля формы. Может случиться, что имя, под которым вы хотите зарегистрироваться, уже занято. Выберите себе новое уникальное имя для регистрации.



Как просматривать, скачивать и видоизменять проекты сообщества?

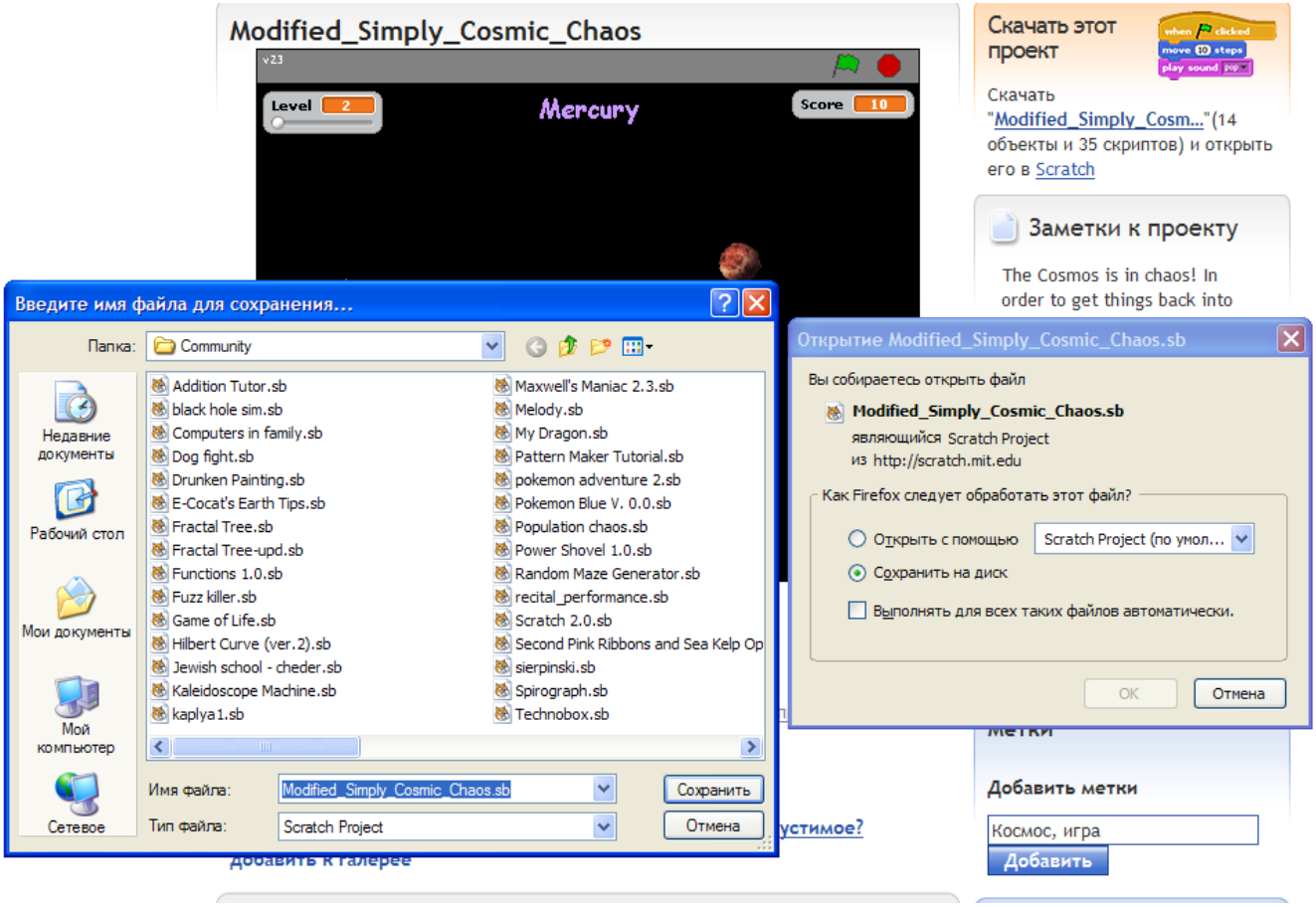


Что мы можем делать с проектами на сайте сообщества:

1. Посмотреть проект в действии – для этого надо просто подождать и проект прямо с сайта загрузится в такой форме, что вы сможете увидеть не только анимацию, но и участвовать в игре, управлять объектом при помощи мышки или специальных клавиш клавиатуры.
2. Пометить проект метками, чтобы потом его было легче найти. Например, в данном случае мы можем добавить к проекту метки «космос», «игра», «астрономия» и т.д.

3. Скачать проект (возможность отмечена в правом верхнем углу)

Посмотреть проект может любой желающий. Скачать проект может только член сообщества. Если вы уже зарегистрировались, то после просьбы скачать проект появится предложение открыть проект в Скретче или сохранить проект на диск. Советуем завести себе отдельную папку для примеров чужих проектов и складывать их туда.

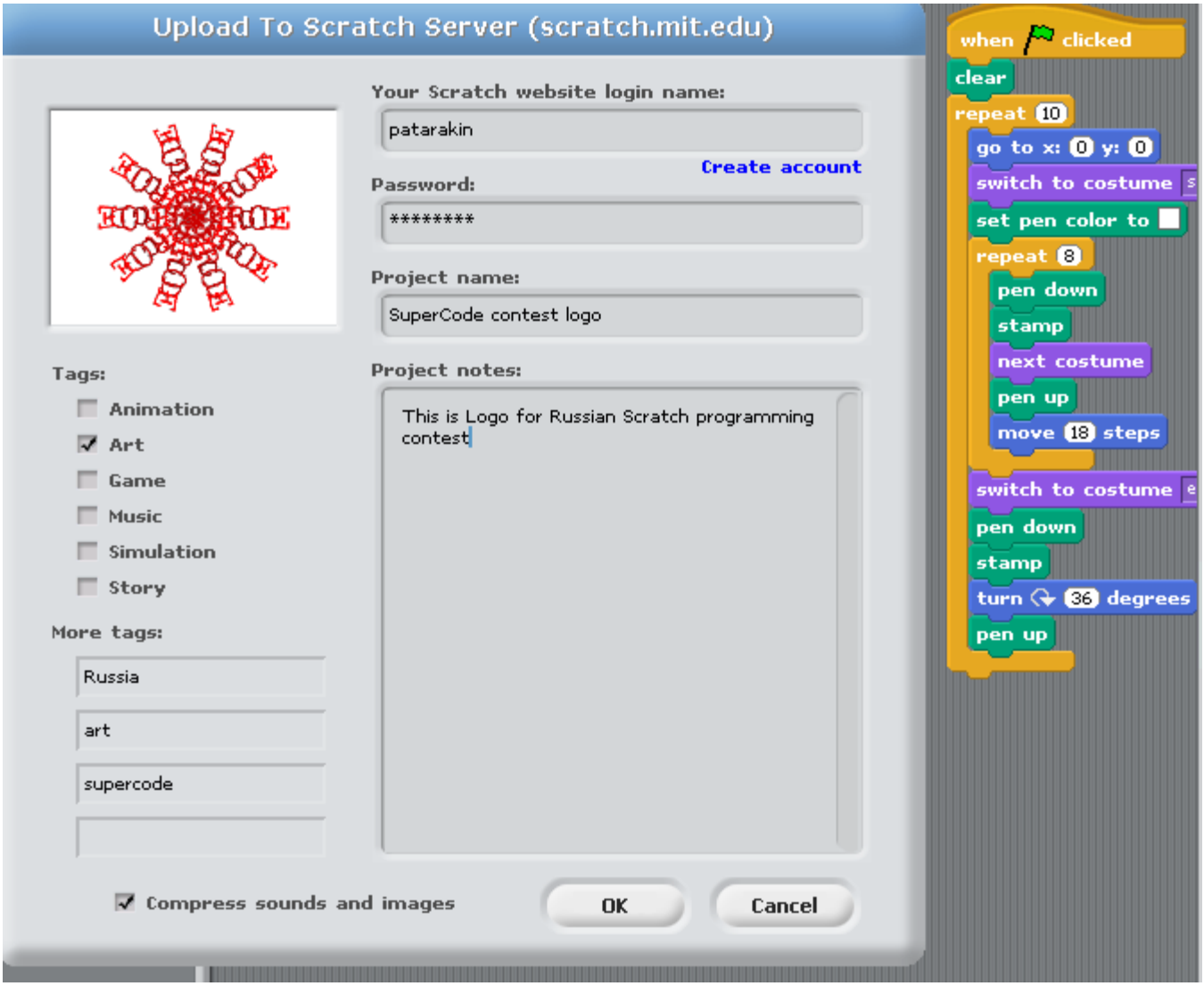


После того как мы сохранили проект, мы можем посмотреть на его составные блоки.

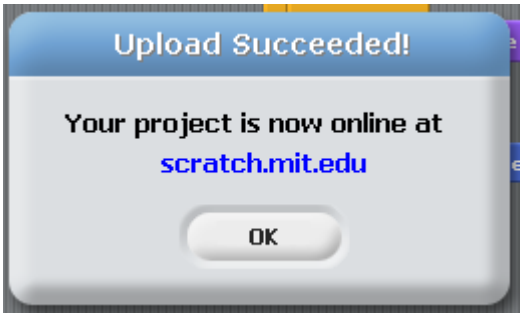
Как можно пополнить архивы сообщества Скретч своими проектами?

Мы сделали Скретч-проект СуперКодЛого и хотим загрузить его на сайт сообщества http://scratch.mit.edu

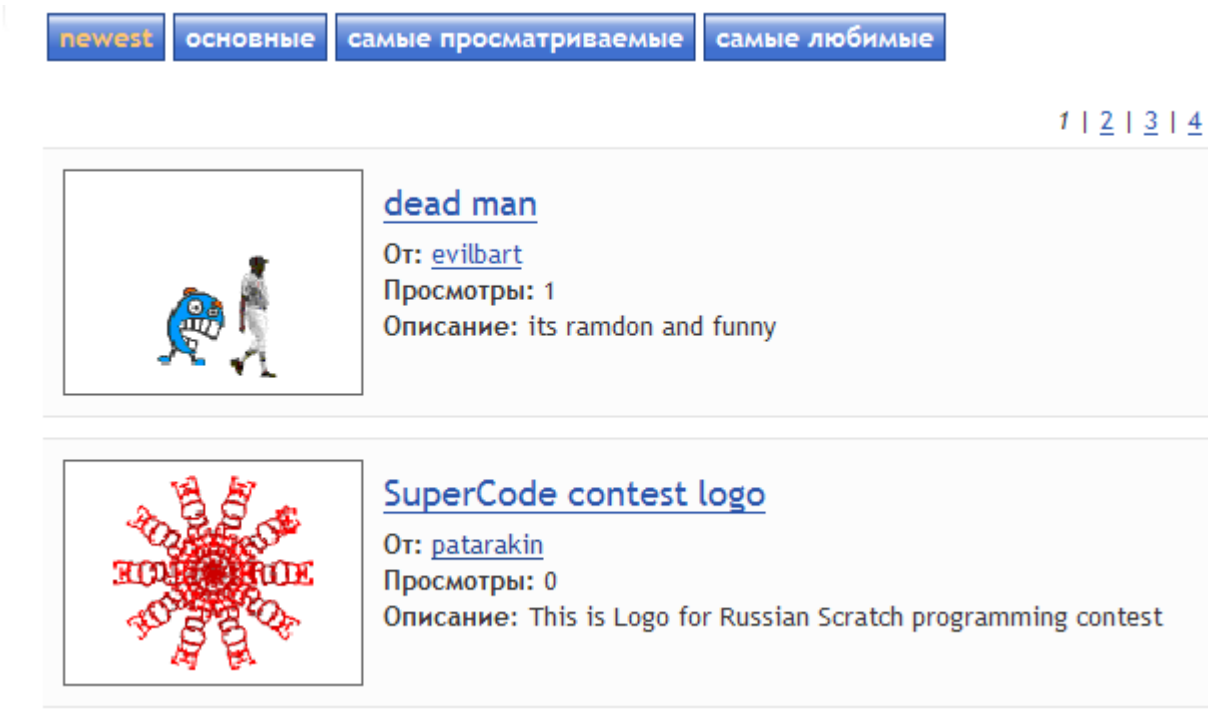
Сохраним проект у себя на машине и положим его на общий сайт сообщества Скретч. Необходимо дать название проекту, указать имя автора - участника сообщества, пароль участника и краткое описание проекта.



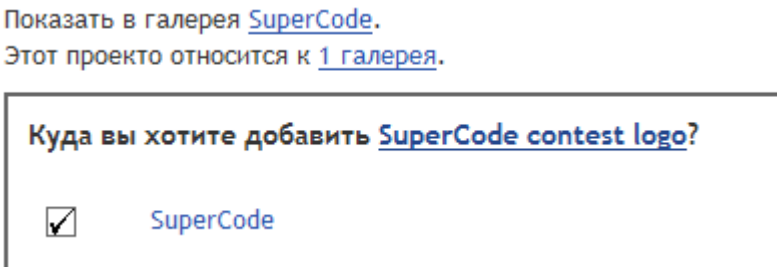
Если все прошло успешно, то появится сообщение, что проект успешно загружен на сайт.



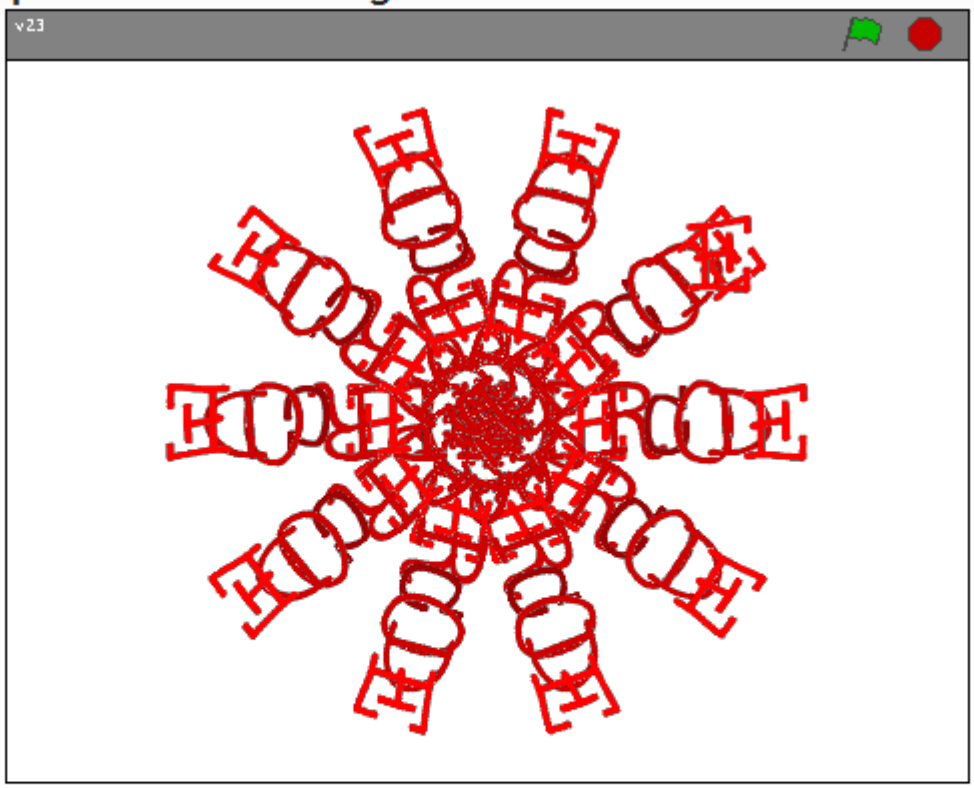
Проект на сайте сообщества:



Отнесем проект в галерею (специальная галерея для российских проектов конкурса Суперкод)



SuperCode contest logo



patarakin опубликован 8 minutes тому назад

Некоторые права сохранены

1 просмотры

Проверить на любимые или метки

Нравится? Добавить к любимым? добавить к галерее

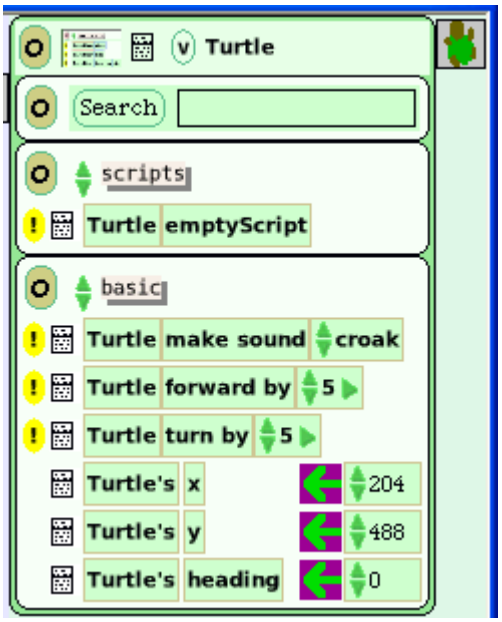
Заключение – что после Скретча?

Скретч является представителем целого направления Squeak, на котором кроме Скретча выросли так же Alice и Croquet. Навыки, полученные учениками в среде Скретч, могут успешно использоваться и в этих средах и среде сетевых сервисов веб 2.0.

Squeak

Squeak – это система создания интерактивных динамичных презентаций. Она ориентирована на детей, и всех, кто хочет получить представление о программировании и конструировании. После запуска программы мы можем нарисовать новый объект. Выбрали режим keep и теперь можем объектом управлять. Навели мышку на центр объекта и получили меню.

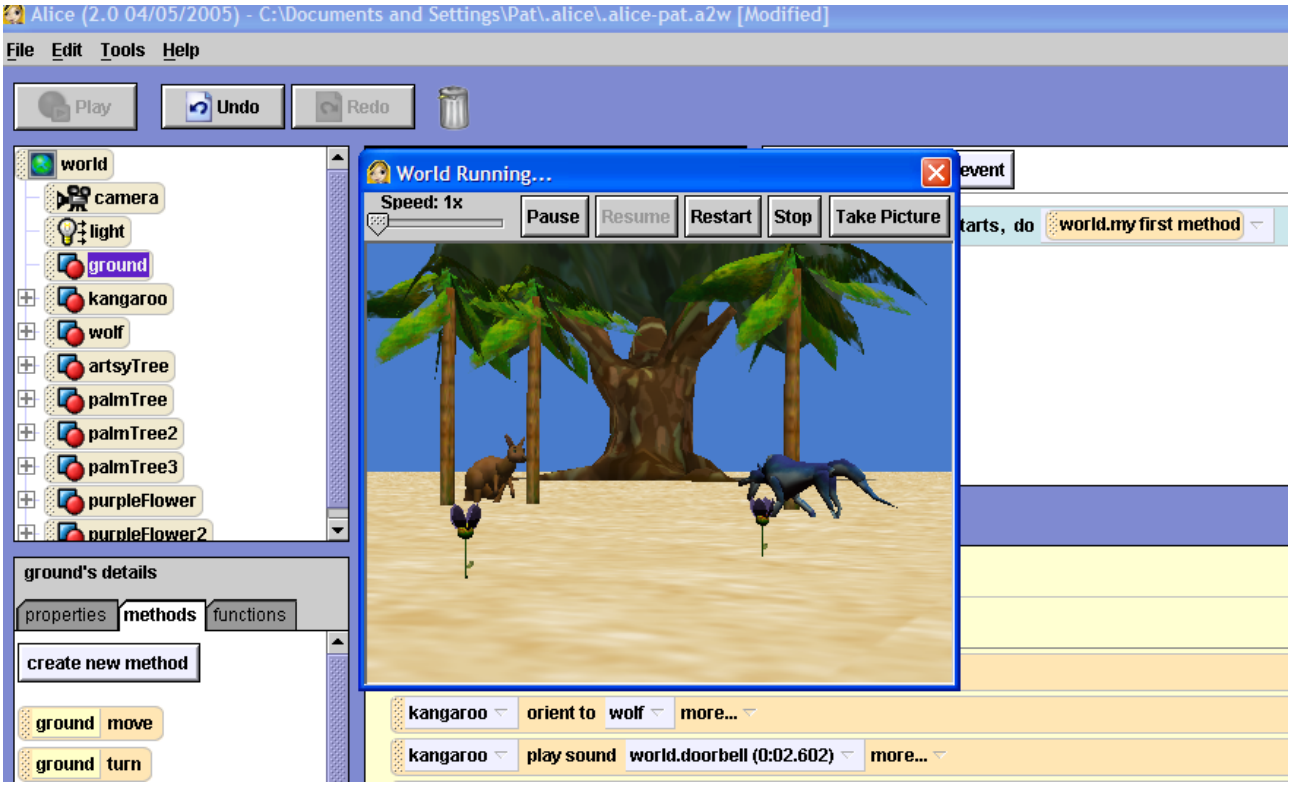
Можем вращать объект, поворачивать, дублировать и писать для объекта управляющие скрипты. Выберем в меню View и добавим в изначально пустой скрипт нужные команды. Команды подобны тем, которые мы встречали в языке Скретч- повернуться, двинуться, издать звук.



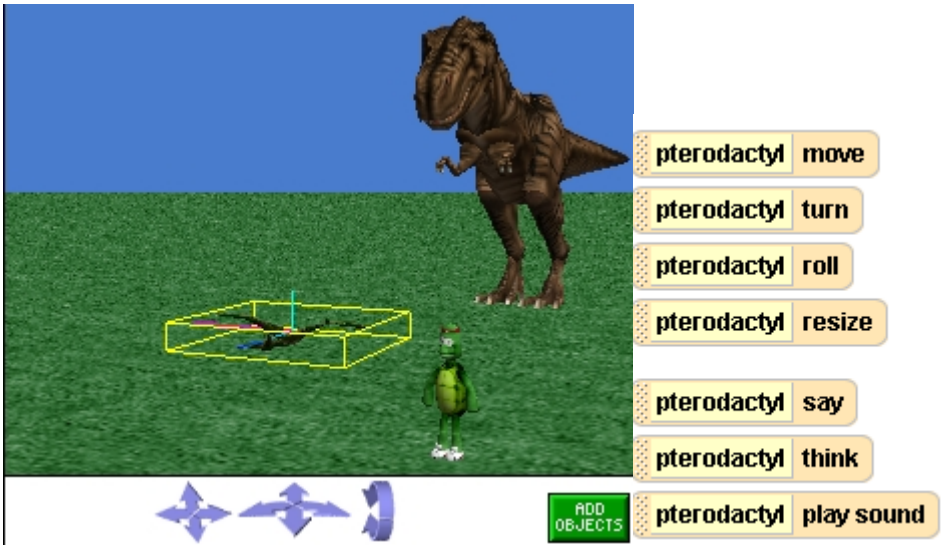
Среда Squeak хорошо подходит для создания систем, в которых множество объектов одновременно выполняют некоторые действия. Например, рыбки плавают в аквариуме, а человечки путешествуют по лесу. Действие каждого из них определяется собственным сценарием, а Squeak следит за одновременной и согласованной работой всех сценариев. Для этого каждый сценарий снабжён таймером.

Alice

Мир программной среды Алиса предназначен для старших школьников и студентов, но устроен во многом подобно среде Скретч. Весь мир раскладывается на объекты и большинство объектов в свою очередь раскладываются на составные кирпичики. Мы можем собирать из простых строительных блоков трехмерный мир, в котором сосуществует множество различных героев. И, как и в среде Скретч, мы можем изучать и копировать строительные блоки, созданные другими людьми.



Мы можем выбрать мир и добавить к нему действующих героев. Поведение каждого героя описывает простыми командами. Например, представленный на следующем рисунке птеродактиль может управляться командами, к которым мы уже привыкли в среде Скретч: иди, повернись, измени размер, скажи, подумай, издай звук и т.п.

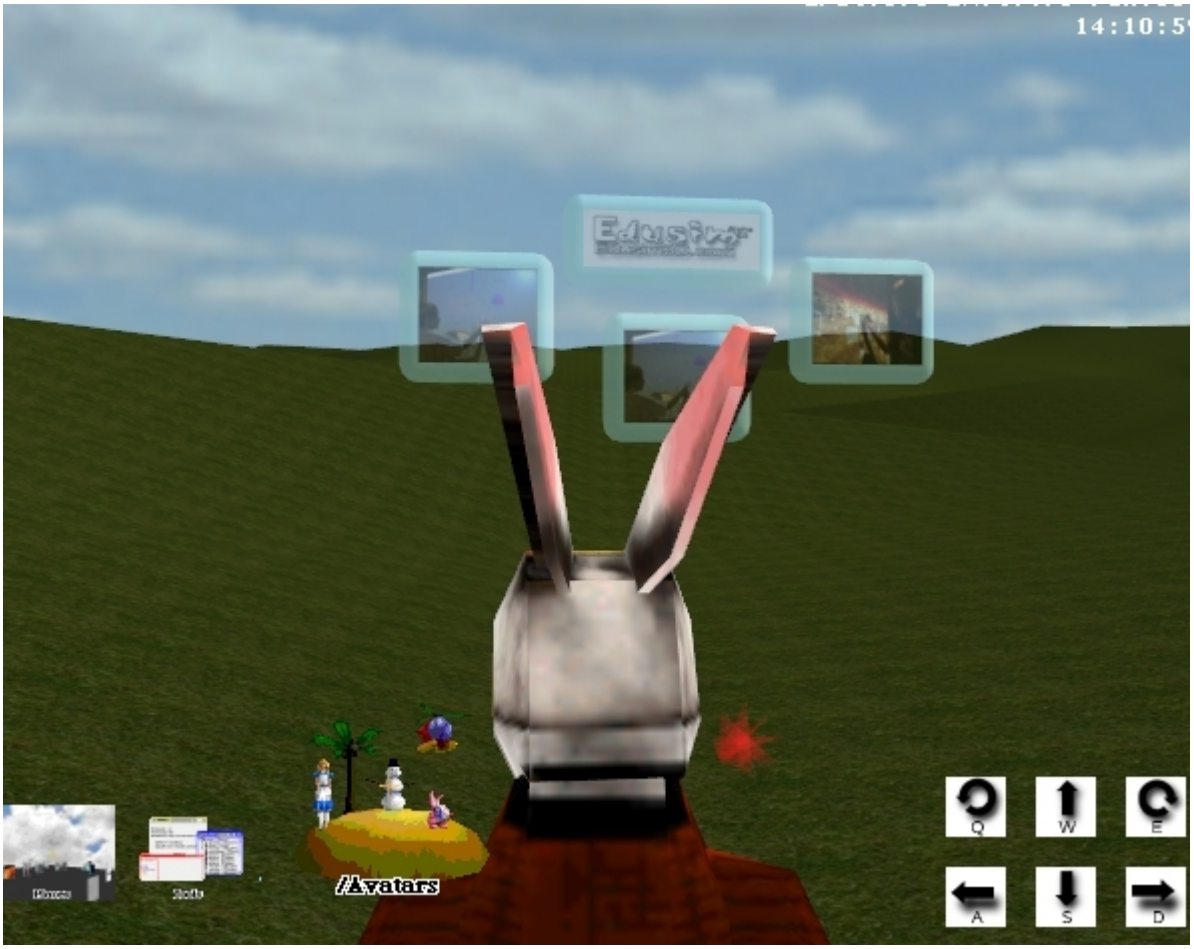


Модели в Алисе более сложные и трехмерные, но после Скретча принципы их создания кажутся очень понятными.

Крокет

http://croquetproject.org/

Croquet - это среда, построенная на базе Squeak. Она предназначена для эффективной организации коллективного общения на базе концепций виртуального присутствия. Виртуальный мир Крокета - без границ и произвольно расширяем и масштабируем. В нем можно создавать и модифицировать все что угодно - вместе с другими людьми; нет разделения между средой разработки и пользовательским миром.



Этот зайчик и есть я - мой аватар в виртуальном мире Крокета..

NetLogo

Близкое Скретчу направление – язык NetLogo. Надо отметить, что идеологом и того и другого направления является Мич Резник. Уже середине 90-х годов Мич Резник и его группа разработали продукт StarLogo, в котором действовали множество черепашек. В конце 90-х Ури Виленский отделился от StarLogo с отдельным проектом NetLogo, который наследовал многие свойства StarLogo. Домашняя страничка проекта - http://ccl.northwestern.edu/netlogo/

Среда NetLogo благоприятна для существования сообщества, поскольку всегда позволяет посмотреть текст программы, написанной другим участником; заимствовать фрагмент программы или отдельную процедуру; видоизменить процедуру по своему усмотрению и использовать ее в рамках другой программы. Одна из наиболее известных и часто демонстрируемых моделей стайного поведения это модель формирования стаи птиц или рыб. В начале все агенты двигаются в разные стороны, но постепенно под действием окружающих из множества агентов формируется стая, во главе которой

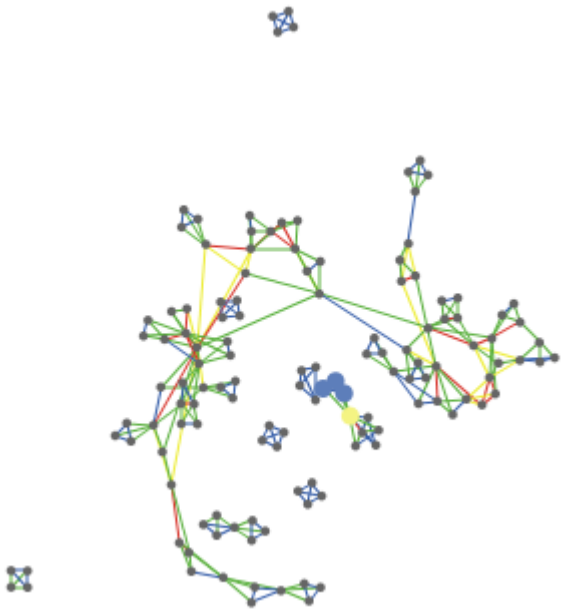
находится птица, рыба или черепашка – вожак. Мы приведем здесь маленький фрагмент программы, управляющий формированием стаи.

```
to flock
find-flockmates
if any? flockmates  [[ align cohere ]]  fd 1
end
```

В переводе с языка NetLogo на русский данное выражение означает, что каждый агент поискал соседей и, если нашел, то скорректировал траекторию своего поведения в ту сторону, куда двигаются окружающие. Язык NetLogo создавался для того, чтобы помогать ученикам размышлять о децентрализованных процессах. Создавая модель стаи птиц, ученики узнают больше о главном правиле сетевых сообществ – «простые правила – сложное поведение». Библиотека моделей, созданных в среде NetLogo обширна и пополняется не только разработчиками, но и членами сообщества -

http://ccl.northwestern.edu/netlogo/models/community/

После того как ученики освоили Скретч и столкнулись с его ограничениями, переход к NetLogo может оказаться очень выигрышным. В среде NetLogo ученики могут создать гораздо более сложные программы, чем в среде Скретч. Полученные в среде Скретч навыки, умение посмотреть, скопировать и видоизменить программу пригодятся ученикам при освоении NetLogo. На следующем рисунке представлена модель формирования творческих групп.



Скретч и веб 2.0

Скретч приучает нас собирать проект из кирпичиков и делиться результатами своих действий с другими людьми. Эти навыки важны не только внутри внутри специальных сред программирования, но и в современных сетевых сообществах. Единство процессов создания, поиска и хранения информационных кирпичиков все чаще можно наблюдать на страницах современных сайтов, использующих концепцию Веб 2.0. Метафора маленьких кирпичиков, из которых дети и взрослые могут собрать простые и очень сложные конструкции, присутствует не только в учебных проектах, но и в большинстве современных сетевых сервисов формата Веб2.0, предназначенных для поддержки организаций и сетевых сообществ обмена знаниями. Например, системы пользовательского поиска позволяют настраивать поисковую систему, указывая свои предпочтения в именах серверов и в ключевых словах. Мы можем не просто наблюдать работу поисковой системы, но настраивать ее, копировать действия другого человека и использовать знания, которые закладывает в поисковый механизм человек, разбирающийся в поиске лучше нас

Другим примером технологии сотрудничества может служить ВикиВики известная школьникам и учителям по проекту «Летописи.ру» - http://letopisi.ru . ВикВики является сетевой самоорганизующейся системой, и может быть использована для освоения стратегий экологического мышления. Каждый член авторского коллектива в среде ВикиВики может выполнять свои простые операции. Чем проще правила индивидуального поведения, тем более сложный характер имеет поведение всей группы. Мы должны не только дать участникам проекта простые и ясные правила, но и объяснить, почему эти правила используются, и показать к каким результатам ведет использование правил и пренебрежение правилами. В учебной ситуации важно, чтобы объяснения были наглядны и соотносились с привычными и повседневными для участников метафорами. В качестве таким наглядных метафор при представлении проекта «Летописи» мы используем метафору строительных кирпичиков конструктора Лего или Скретч, из которых можно собрать здание, машину или программу, управляющую поведением машины. Возможность использовать страницы ВикиВики как строительные блоки наиболее полно проявляется благодаря встроенному в МедиаВики механизму шаблонов. Шаблоны МедиаВики - особые страницы, содержимое которых можно вставить в другие страницы.

Изменения в шаблоне отражаются на всех страницах, в которые они встроены. Шаблоны позволяют создавать образцы и повторно использовать их на страницах.

Чаще всего внимание обращают на то, что в ВикиВики очень просто и очень быстро можно создать и разместить в сети новую страничку. При этом из зоны внимания выпадает то, что среда ВикиВики сильна именно взаимосвязями страниц и коллективностью усилий. То, что создано одним человеком, может быть в дальнейшем использовано другими людьми. Когда ученик или учитель добавляет в ВикиВики новую статью или новую фотографию, то эта фотография или статья перестает быть их собственностью. Теперь она является общим достоянием и может быть включена в состав новых статей.

Кирпичики Лего или кирпичики Скретч полезны и сильны не тем, что они просты, а тем, что для них определен точный стандарт и они всегда совместимы друг с другом. Точно так же статьи внутри ВикиВики обретают мощь тогда, когда при их изготовлении мы следуем определенным стандартам. Мы можем собирать статьи из готовых блоков так же, как собирается программа из кирпичиков в языке Скретч.

Метафора строительных кирпичиков находит воплощение не только в поиске и хранении материалов, но и в планировании деятельности. Например, планирование совместной деятельности в календарях Google выглядит как заполнение дней и часов кирпичиками событий. Каждый пользователь может создать множество разноцветных календарей и относить каждое событие к календарю определенного цвета. Кроме того, пользователи могут открывать доступ к своим календарям другим людям и тогда заполнение времени превращается в увлекательную игру с кирпичиками событий.

Все перечисленные в руководстве учебные среды являются свободными программами. Ученики и учителя могут использовать среды и проекты по своему усмотрению. И это еще одна важная особенность Скретча – он постепенно приучает нас работать вместе и делиться результатами своей деятельности

Литература

1. Bruckman, A., 1998, Community support for constructionist learning. Computer supported collaborative work. The Journal of Collaborative Computing, 7, 47-86.
2. Kay A. 2000, Sqeak in School http://www.squeakland.org/learn/index.html
3. Kay A., Predicting The Future. Stanford Engineering, Volume 1, Autumn 1989, pg 1 – 6.
4. Kay A., 1991, Computers, Network and Education. Scientific American, September, pp., 138 - 148.
5. Kafai, Y.B., Peppler, K.A., & Chin, G. (2007). High Tech Programmers in Low Income Communities: Creating a Computer Culture in a Community Technology Center. In C. Steinfeld, B. Pentland, M. Ackermann, & N. Contractor (Eds.), Proceedings of the Third International Conference on Communities and Technology (pp. 545-562). New York: Springer.
6. Malan D. J., Leitner H., *Scratch for Budding Computer Scientists*. 38th ACM Technical Symposium on Computer Science Education. Covington, Kentucky. March 2007.
7. Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., and Resnick, M. (2004). Scratch: A Sneak Preview. Second International Conference on Creating, Connecting, and Collaborating through Computing. Kyoto, Japan, pp. 104-109.
8. Papert S, Caperton G., 1999, Vision for Education: The Caperton-Papert Platform http://www.papert.com/articles/Vision_for_education.html
9. Papert S., 1993, The children's machine: rethinking school in the age of the computer BasicBooks.
10. Papert S., 2000, What's the big idea: Towards a pedagogy of idea power. IBM Systems Journal, vol., 39, no., 3-4. http://www.research.ibm.com/journal/sj/393/part2/papert.html
11. Papert, S., Resnick M., 1995. "Technological Fluency and the Representation of Knowledge". Proposal to the National Science Foundation. MIT Media Laboratory. Cambridge, MA.
12. Peppler, K. A. & Kafai, Y. B. (2007). What video game making can teach us about learning and literacy: Alternative pathways into participatory cultures. Paper to be presented at the Digital International Games Research Association meeting in Tokyo, Japan.

13. Peppler, K. & Kafai, Y. B. (2007). From SuperGoo to Scratch: Exploring Media Creative Production in an Informal Learning Environment. Journal on Learning, Media, and Technology, 32(2), 149-166.
14. Resnick, M., 2003, Thinking Like a Tree (and Other Forms of Ecological Thinking). International Journal of Computers for Mathematical Learning, vol. 8, no. 1, pp. 43-62
15. Resnick, M.,1994, Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds. Cambridge, MA: MIT Press.
16. Resnick M., 1999, Decentralized Modeling and Decentralized Thinking Published in: Modeling and Simulation in Science and Mathematics Education (pp., 114-137), edited by W. Feurzeig and N. Roberts. Springer: New York., 1999
17. Resnick, M., 1996. "Towards a Practice of 'Constructional Design'." In Innovations in Learning: New Environments for Education, edited by L. Shauble and R. Glaser. NJ: Lawrence Erlbaum. Hillsdale.
18. Senge P., 1998, Systems Change in Education, Education for Meaning and Social Justice, Volume 11, Number 3, September 1998,
19. Senge P., 2000, "A World Shaped by Choice", Imagine, essays on the future edited by Marianne Williamson, Rodale Press, fall 2000.
20. Бобровский С. История о том, как я писал историю для начинающих изучать Squeak с нуля http://sbobrovsky.livejournal.com/167512.html
21. Бобровский С. История о том, как я писал историю о построении трёхмерных миров в Alice. http://sbobrovsky.livejournal.com/170524.html
22. Бобровский С. История о том, как я писал историю про быстрый старт в Croquet http://sbobrovsky.livejournal.com/168387.html
23. Патаракин Е.Д., Ярмахов Б.Б., 2006, NetLogo в моделировании социальных феноменов и на занятиях по менеджменту // Конференция «Свободное программное обеспечение в высшей школе», Переславль-Залесский, 28–29 января 2006 года, стр. 36 – 38
24. Патаракин Е.Д., Ярмахов Б.Б., 2007, Веб 2.0 Управление, изучение и копирование, Educational Technology & Society - 10(2) 2007, ISSN 1436-4522 http://ifets.ieee.org/russian/depository/v10_i2/html/2.htm
25. Патаракин Е.Д. Освоение медиа-культуры через учебные игры с маленькими кирпичиками знаний. В книге Судьба России: вектор перемен, 2007, ISBN: 978-5-8291-0904-2