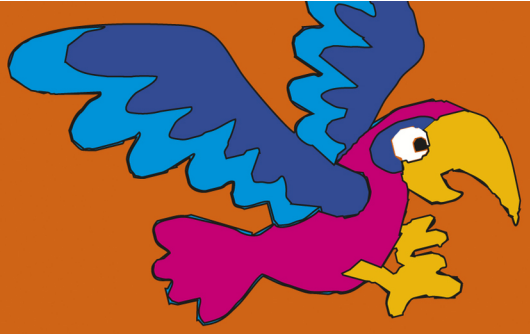
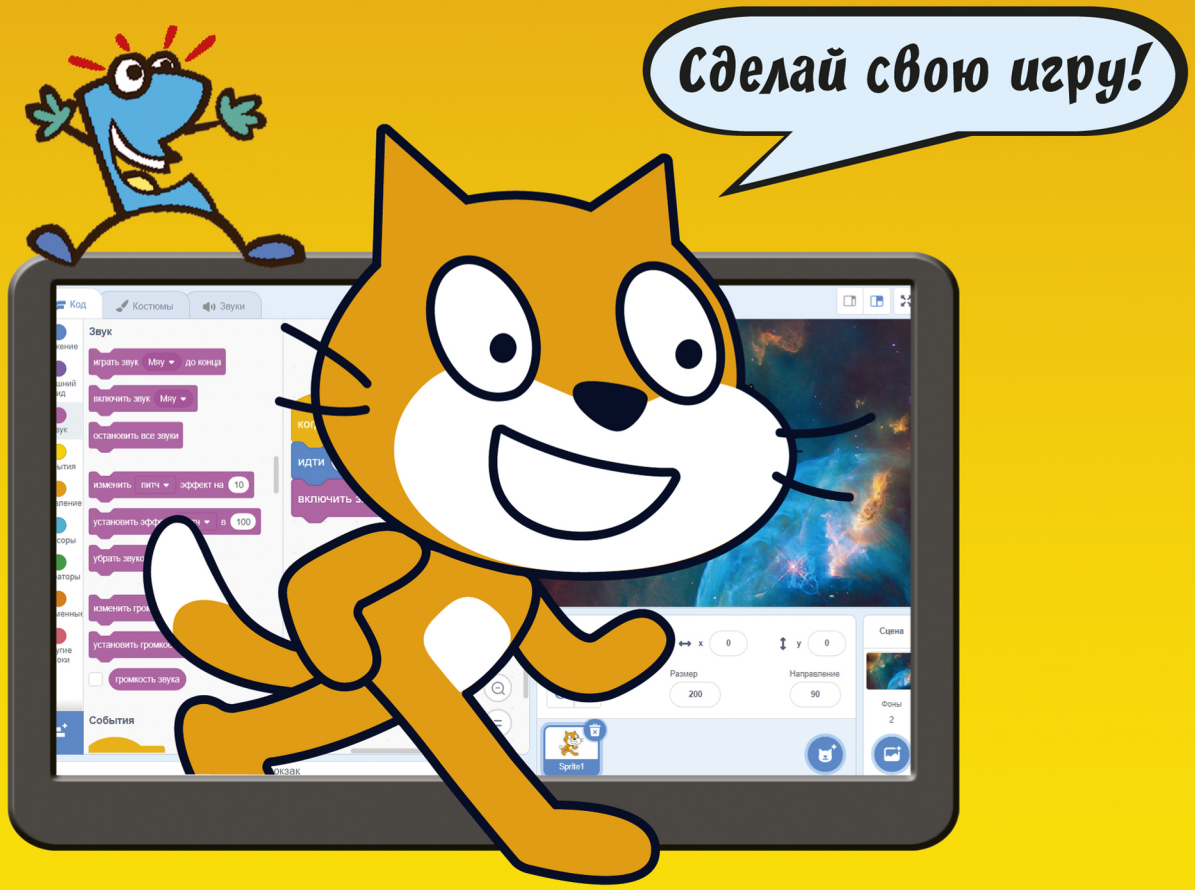


bhv®

Денис Голиков

Scratch 3

для юных программистов



ДЕНИС ГОЛИКОВ

SCRATCH 3 ДЛЯ ЮНЫХ ПРОГРАММИСТОВ

Санкт-Петербург
«БХВ-Петербург»
2020

УДК 004.43-053.2
ББК 32.973.26-018.1
Г60

Голиков Д. В.

Г60 Scratch 3 для юных программистов. — СПб.: БХВ-Петербург, 2020. — 168 с.: ил.

ISBN 978-5-9775-6591-2

Книга написана на основе опыта обучения программированию на языке Scratch 3 в кружке юных программистов и протестирована на сотне детей 7–12 лет. Материал рассчитан на самостоятельное, без помощи взрослых, изучение Scratch 3 школьниками 2–5 классов, имеющими базовые навыки управления компьютером. Доходчивость изложения позволит детям сразу начинать создавать увлекательные проекты, а присущий автору юмор сделает это занятие веселым. Большое количество проектов и заданий для креативной самостоятельной работы поможет творчески применять многочисленные возможности Scratch 3.

Юные программисты узнают о логических и математических операторах, циклах и условиях, научатся создавать забавные рисунки и узоры, музыкальные проекты, мультфильмы и веселые игры, которые будут работать на всех устройствах — и на смартфонах, и на планшетах, и на компьютерах.

Для детей младшего и среднего школьного возраста

УДК 004.43-053.2
ББК 32.973.26-018.1

Группа подготовки издания:

Руководитель проекта	*Евгений Рыбаков*
Зав. редакцией	*Екатерина Сависте*
Компьютерная верстка	*Ольги Сергиенко*
Дизайн обложки	*Карины Соловьевой*

«БХВ-Петербург», 191036, Санкт-Петербург, Гончарная ул., 20.

ISBN 978-5-9775-6591-2

ВВЕДЕНИЕ ДЛЯ ВЗРОСЛЫХ

Дорогие друзья! (Сейчас я обращаюсь к детям.) Введение можете не читать, переходите сразу к главе 1, начинайте делать весёлые мультики. А родителям будет полезно узнать некоторые сведения о Scratch.

Целью книги является наглядное обучение программированию школьников младших классов. Книга написана на основе опыта обучения программированию на Scratch в кружке юных программистов и протестирована на сотне детей 7–12 лет.

Материал рассчитан на самостоятельное, без помощи взрослых, изучение Scratch школьниками 1–5 классов, имеющими базовые навыки управления компьютером. Дети должны уметь пользоваться мышью, запускать программы, щелкая по их ярлыкам, и т. п., а также считать, умножать и делить. Более сложные математические понятия (отрицательные числа, десятичные дроби, проценты, оси координат, градусы) объяснены на страницах книги.

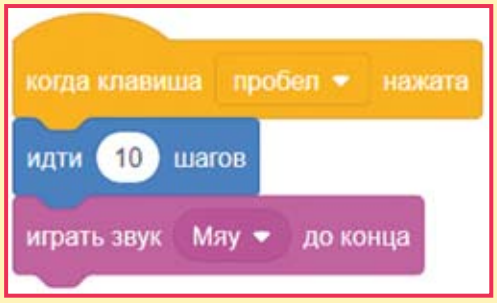
Особенность книги — очень подробное пошаговое описание процесса создания программ. Все остальные книги опускают многие стороны процесса, подразумевая, что читатель сам догадается о мелких деталях. Здесь же скриншотами (экранными снимками) представлен процесс создания проектов целиком. Все вопросы, возникающие у детей, были сняты в ходе тестирования книги в кружке. Принцип обучения такой: сначала конструируем сложную и непонятную программу (именно конструируем, так как процесс программирования в Scratch подобен созданию моделей из деталей конструктора), потом запускаем её и пытаемся немного изменить. Наблюдая за сделанными изменениями, начинаем понимать, как программа работает. В книге нет никакого введения, дети сразу начинают делать весёлые мультики, а потом даже игры. Одна из главных целей — постараться, чтобы ребёнку не было скучно, поэтому шутки и юмор для игр добавлял мой сын пятиклассник.

Что такое Scratch?

Scratch — это визуальный язык программирования, в котором программа складывается из разноцветных блоков. Детям ничего не нужно писать, как в других языках программирования. Блоки имеют защёлки, которые не позволяют соединить несовместимые блоки.

Талисманом Scratch является симпатичный рыжий Кот. Он встречает всех, открывших редактор.

Знакомство с программированием начинается с создания простейших программ, например таких, как вот эта.

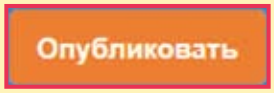


При нажатии на клавишу <Пробел> эта программа передвинет Кота на 10 шагов и проиграет звук «Мяу».

Scratch работает в браузере, поэтому программировать можно и на компьютере, и на планшете, а запускать готовые проекты, сделанные детьми (мультфильмы и игры), можно даже на смартфоне или телевизионной приставке!

Как поделиться проектом, созданным на Scratch?

Зарегистрируйтесь на сайте https://scratch.mit.edu/, создайте проект, откройте доступ к нему, нажав кнопочку **Опубликовать** в строке меню.



Скопируйте ссылку на проект в строке адреса https://scratch.mit.edu/projects/14155407/ и поделитесь ею в Интернете. Также скопировать ссылку можно, нажав кнопку **Copy Link** на странице проекта.

Кто создал Scratch?

Проект по созданию Scratch инициирован в 2003 г. при финансовой поддержке компаний Science Foundation, Intel Foundation, Microsoft, MacArthur Foundation, LEGO Foundation, Code-to-Learn Foundation, Google, Dell, Fastly, Inversoft и MIT Media Lab research consortia.

Scratch создан в лаборатории Lifelong Kindergarten Массачусетского технологического института под руководством профессора Митчела Резника (Mitchel Resnick) в 2007 г.

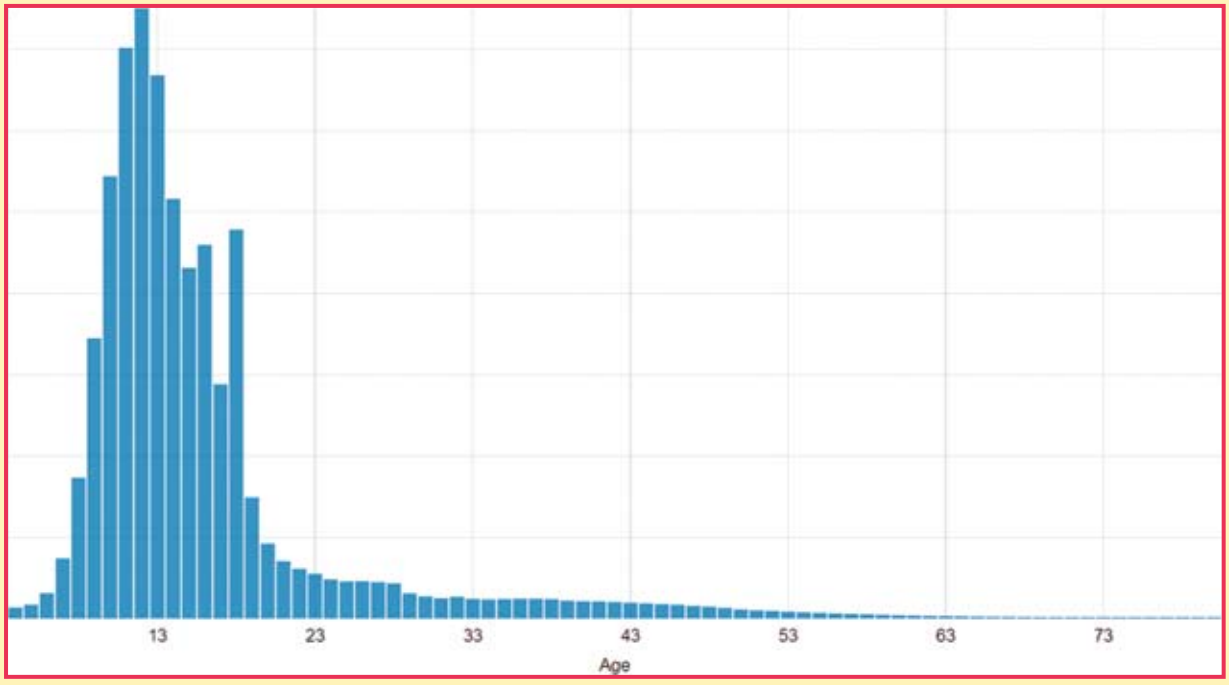
Познакомиться с командой разработчиков Scratch вы можете на странице

https://scratch.mit.edu/info/credits/

На какой возраст рассчитан Scratch?

Создатели Scratch разрабатывали его специально для детей 8–16 лет. Однако 6–7-летние дети, которые умеют читать, считать, а также пользоваться мышью, тоже могут создать простые проекты.

Основной возраст участников сообщества — 8–18 лет.



Удивительным открытием для меня стал факт, что креативные пенсионеры обожают создавать проекты на Scratch. Оказалось, это отличная гимнастика для ума и весёлый способ провести свободное время!

Насколько популярен Scratch?

На сайте http://scratch.mit.edu зарегистрировано более 40 млн пользователей со всего мира. Из них 15 млн из США, 2,4 млн из Великобритании и всего 0,009 млн из России.

Подробную статистику о Scratch можно посмотреть на странице https://scratch.mit.edu/statistics/

Где найти Scratch?

Существует два способа работы в среде Scratch. Самый простой способ — работа в онлайновом редакторе Scratch, который можно запустить по адресу:

https://scratch.mit.edu/projects/editor/

Для того чтобы иметь возможность сохранять созданные проекты, необходимо зарегистрироваться.

Второй способ — работа в офлайновом редакторе, который можно скачать со страницы

https://scratch.mit.edu/download

Существуют версии под Windows и Mac OS.

Где можно использовать Scratch?

Программирование на Scratch — очень весёлое занятие, поэтому лучше всего заниматься им в группах, тогда дети смогут сразу делиться своими проектами, обсуждать их, совместно придумывать сюжеты.

Scratch идеально подходит для использования на дополнительных уроках в начальных классах (в группах продлённого дня). Дети очень увлекаются созданием проектов, благодаря чему их поведение улучшается.

Scratch можно использовать в библиотеках, оборудованных компьютерами. Там дети могут создавать проекты о героях прочитанных книг, работать совместно.

Scratch хорошо подходит для организации кружков юных программистов на базе учреждений дополнительного образования.

При использовании Scratch дома желательно зарегистрироваться на сайте и размещать все проекты там. На сайте есть большое русскоязычное сообщество, в котором дети смогут найти единомышленников, задавать вопросы и обсуждать проекты.

Где найти дополнительную информацию о Scratch?

Дополнительная информация о Scratch на русском языке:

- на странице официального форума по адресу https://scratch.mit.edu/discuss/27/;
- в ScratchWiki по адресу http://scratch-wiki.info/;
- в Википедии;
- на сайте http://scratch4russia.com/.

О книге

О чём узнают дети, прочитавшие эту книгу?

Дети узнают о том, что такое цикл, условный блок, цикл с условием, логическое выражение, координатная плоскость, процент, десятичная дробь, градус, переменная, список.

Чему научаться дети, прочитавшие книгу?

Дети научаться создавать мультфильмы, игры, сложные скрипты (то есть программы), рисовать в векторном и растровом графических редакторах, изменять звук, вводить, выводить и обрабатывать информацию.

Правила работы с книгой

Книга состоит из 19 глав. Создание проектов разбирается подробно, по шагам, с объяснением новых понятий и блоков. В конце каждой главы приведены задания для самостоятельного выполнения. Главы нужно изучать последовательно, одну за другой, иначе можно пропустить объяснение важных понятий. Будет лучше, если все созданные проекты ваш ребёнок станет выкладывать на сайте http://scratch.mit.edu. В этом случае я смогу ответить на вопросы и проверить выполнение заданий. Обязательно добавьте меня в друзья на этом сайте. Мой профиль:

https://scratch.mit.edu/users/scratch_book/

Не забудьте подтвердить свой электронный адрес (e-mail) после регистрации!

Условные обозначения

Жирным шрифтом выделены элементы интерфейса программы Scratch.

Названия блоков выделены узким шрифтом.

Названия переменных и списков выделены **узким жирным шрифтом**.

Названия клавиш клавиатуры заключены в угловые скобки, например <Пробел>.

Установка Scratch

Если вы решили использовать офлайн-версию программы, прежде всего помогите своим детям установить её на компьютер. Для этого перейдите по ссылке https://scratch.mit.edu/download, скачайте и установите Scratch Offline Editor.

Об авторе

Голиков Денис Владимирович — Scratch-пропагандист. Окончил Московский энергетический институт по специальности «Промышленная электроника».

В 2013–2019 гг. педагог дополнительного образования по Scratch. В 2014 г. кружок Scratch награждён премией губернатора Московской области.

В 2015 г. финалист Конкурса инноваций в образовании организованного Институтом образования НИУ ВШЭ при поддержке Агентства стратегических инициатив.

Автор многочисленных учебно-методических комплектов по Scratch, Scratch4Arduino, Arduino, электронике, Интернету вещей и другим темам для детей 7–12 лет.

Автор онлайн-видеокурсов по программированию на Scratch и в Minecraft.

Автор бестселлеров «Scratch для юных программистов», «40 занимательных проектов на Scratch для юных программистов» и «42 занимательных проекта на Scratch 3 для юных программистов».

В настоящее время работает начальником направления «Образование» в частной российской космической компании «Спутникс».

Контакты

- Электронная почта автора scratch.book@ya.ru.
- Сайт автора в Интернете http://scratch4russia.com/.
- Страница автора в Facebook https://www.facebook.com/ScratchBook4u.
- Страница автора в социальной сети «ВКонтакте» http://vk.com/scratch.book.
- Работы автора на сайте Scratch https://scratch.mit.edu/users/scratch_book/.

- Страница автора на портале обучения Scratch http://scratched.gse.harvard.edu/user/21346.
- Онлайн-видеокурсы автора http://codim.online/1.

Благодарности

Огромное спасибо моим детям Артёму и Алисе, которые помогали придумывать игры и шутки для книги.

Выражаю благодарность всем моим ученикам, посещавшим в 2013–2019 гг. «Кружок юных программистов» в г. Химки. Без вас написание этой книги было бы невозможным.

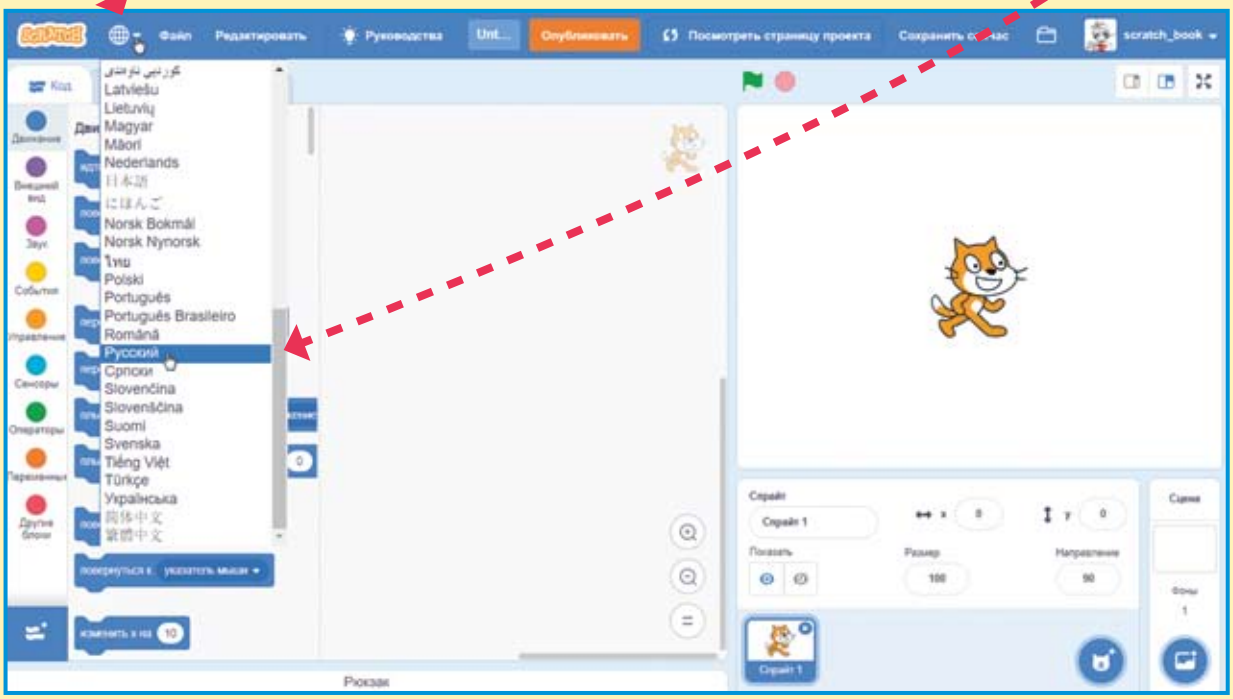
Огромное спасибо коллективу издательства «БХВ-Петербург» и лично Евгению Рыбакову и Анне Кузьминой.

ГЛАВА 1. ЗНАКОМСТВО СО SCRATCH

1.1. Знакомство с интерфейсом

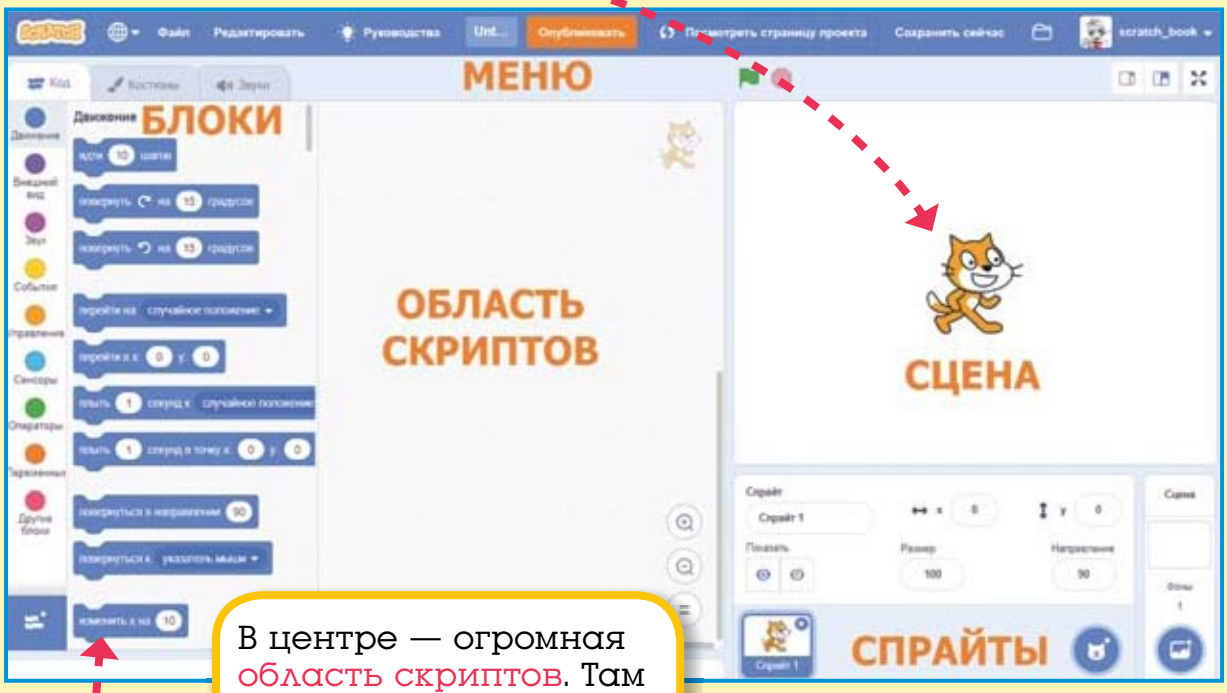
Запустите Scratch — перейдите на сайт https://scratch.mit.edu/ и нажмите кнопку **Создавай**.

Откроется страница онлайн-редактора. Если на странице вы увидите надписи на английском языке, то первым делом надо переключить её на русский интерфейс. Для этого щёлкните на значке глобуса в строке меню, а затем выберите русский язык почти в самом конце списка языков.



Теперь можно осмотреться.

Белое поле справа — это сцена, на ней будет видно, как работает проект. По сцене будут перемещаться спра́йты (персонажи), на ней вы будете рисовать и изменять её фон. Сейчас на сцене всего один спрайт — Кот.



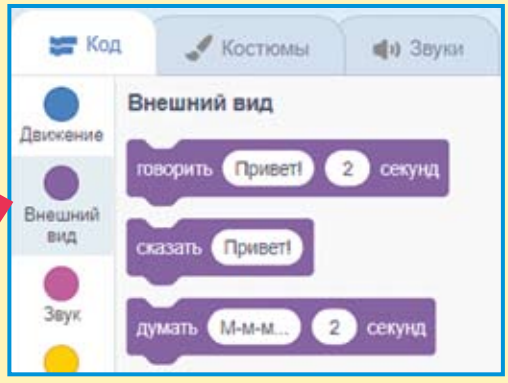
В центре — огромная область скриптов. Там мы будем собирать скрипты проекта из разноцветных блоков, которые хранятся в палитре блоков, расположенной слева.

Все спрайты проекта находятся в области спрайтов, которая расположена под сценой.

Сейчас выбраны синие блоки **Движение**.

Выберите фиолетовые блоки **Внешний вид**.

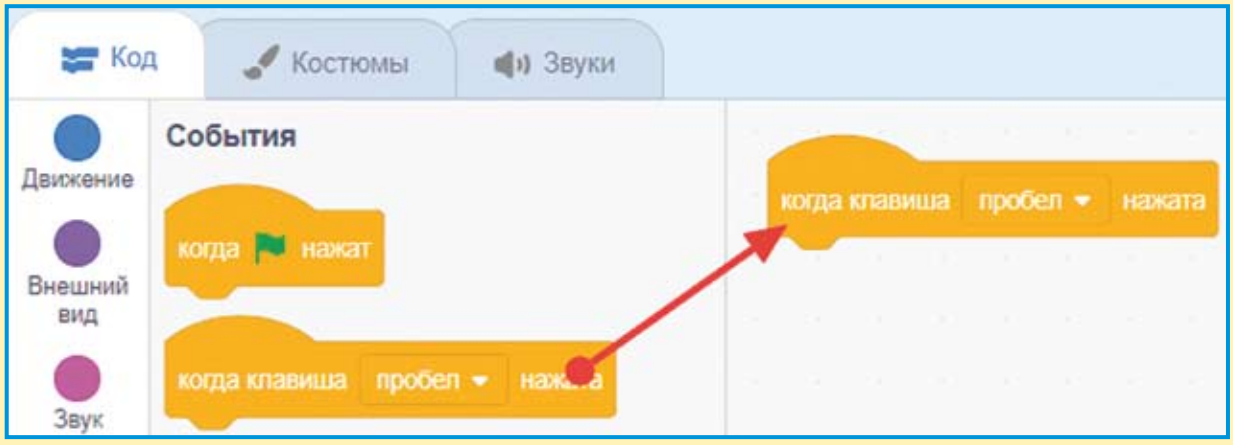
Пощёлкайте по блокам других цветов.



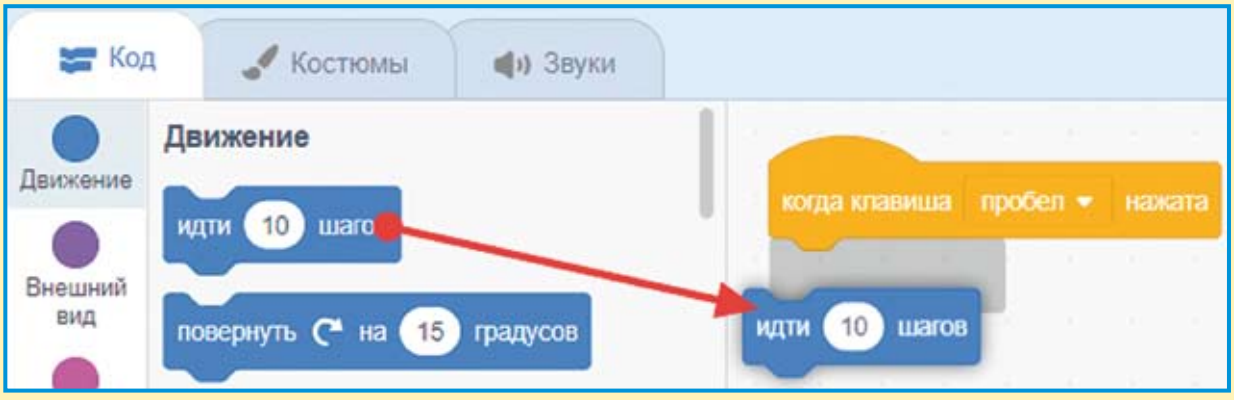
С остальными вкладочками и кнопочками мы познакомимся позднее, а теперь пришло время сделать первый проект!

1.2. Первый проект

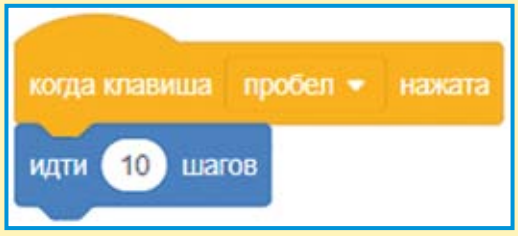
Выберите блоки **События**. Щёлкните мышью на блоке когда клавиша пробел нажата и, не отпуская левую кнопку мыши, тяните его в область скриптов.



Расположите блок в верхней части области скриптов и отпустите левую кнопку мыши. Затем выберите синие блоки **Движение** и вытащите в область скриптов блок идти 10 шагов. Тащите его прямо к первому блоку. Когда он захочет к нему прицепиться, то появится белая полоса, в этот момент отпускайте левую кнопку мыши — блок встанет на место.



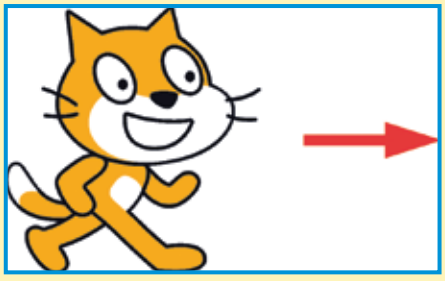
Получилась первая программа, состоящая из одного *скрипта*. Скриптами будем называть кусочки, из которых состоит программа спрайта (персонажа).

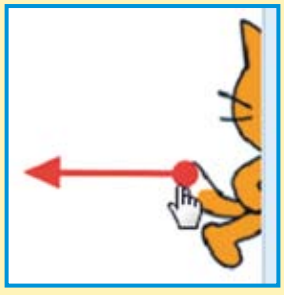


Важно!
Каждый скрипт начинается с блока **Событий** с круглой «шапочкой». Скрипт выполняется сверху вниз. Каждый блок по очереди выполняет своё действие.

Нажимайте клавишу <Пробел> и посмотрите, что будет происходить на сцене. Котик пойдёт направо!

Если нажимать клавишу <Пробел> много раз, то Котик скроется за краем сцены. Вытащите его обратно за хвост.



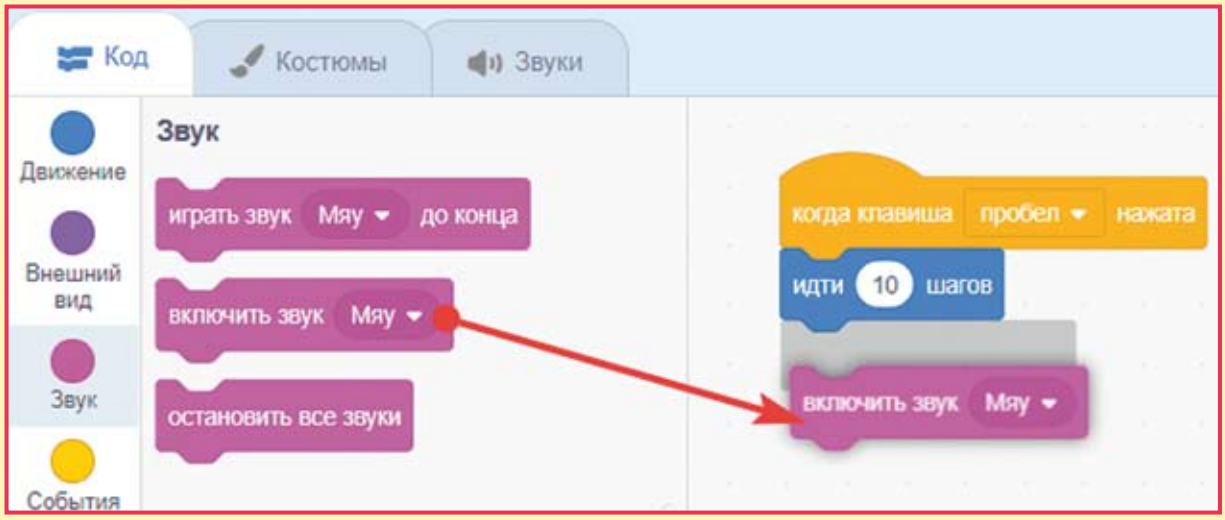


Теперь нажмите клавишу <Пробел> и не отпускайте её — Кот побежит! Снова вытащите его на середину сцены.

Отлично! Бегать Кота научили, теперь научим его мяукать!

1.3. Блоки звука

Выберите сиреневые блоки **Звука**. Прицепите к скрипту блок включить звук Мяу.

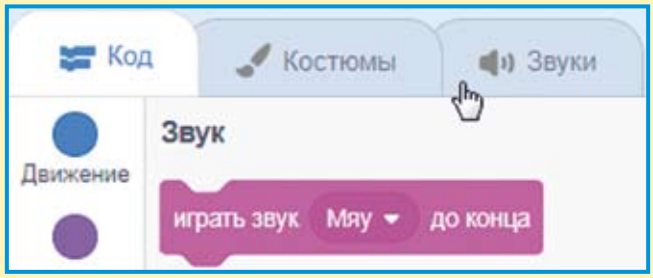


Снова нажимайте клавишу <Пробел> — Кот идёт и мяукает!

Совет

Если Кот не мяукает, то проверьте громкость звука и включите колонки.

Мяукать Кота научили, а теперь попробуем научить его лаять! Щёлкните на вкладку **Звуки**.



Затем наведите мышь на круглый голубой значок слева внизу.

В открывшемся меню нажмите кнопку **Выбрать звук.**

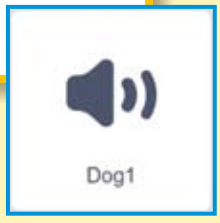
Откроется библиотека звуков. Она содержит более 350 различных звуков, мелодий и эффектов.

Совет

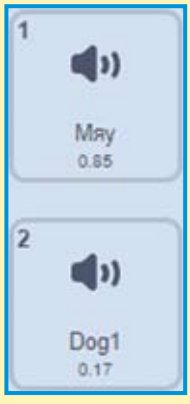
Для того чтобы прослушать звуки, просто наведите на них указатель мышки.

Выберите категорию **Животные**.

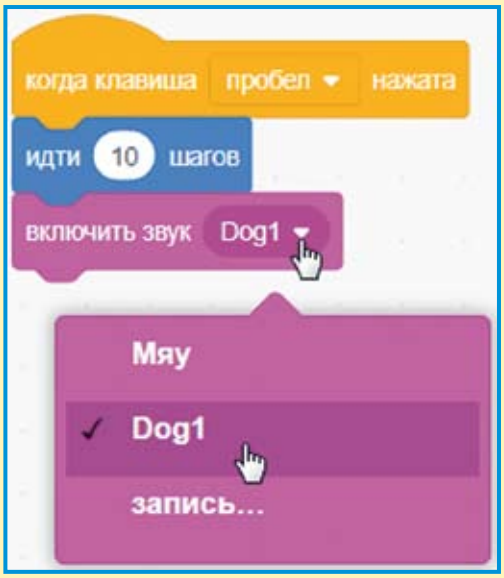
Щелкните по значку звука Dog1.



Теперь Кот может использовать два звука — либо «Мяу», либо «Dog1».



Перейдите на вкладку **Скрипты**. Нажав на белый треугольничек, раскройте список звуков в сиреневом блоке и выберите звук «Dog1».



Нажимайте клавишу <Пробел>, теперь Кот бегает и лает! Так гораздо веселее! Получился Кот-иностранец.

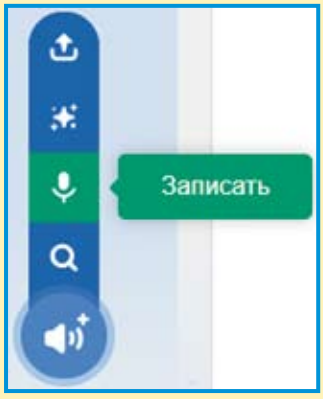
Ну, вот вы и сделали свой первый проект на Scratch, осталось только дать ему нормальное имя и поделиться им. Замените имя проекта «Untitled-1» на «Кот лает».

Затем нажмите оранжевую кнопку **Опубликовать** — откроется страница проекта, на которой можно собирать лайки.

Скопируйте ссылку на проект и поделитесь ею с друзьями. Ссылка выглядит примерно вот так: https://scratch.mit.edu/projects/287345328/ (но числовой номер проекта у вас будет свой).

1.4. Задания

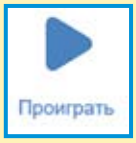
1. С помощью кнопки **Записать** (на вкладке **Звуки**) попробуйте записать звук с микрофона.



Для начала записи нажмите кнопку с кружочком.

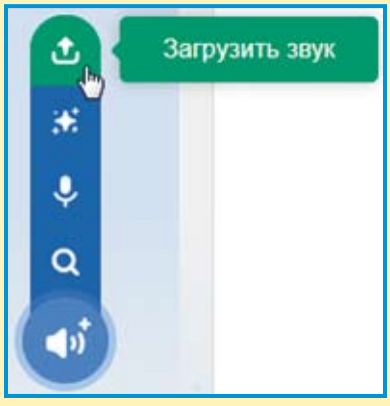
Для остановки записи нажмите квадратную кнопку.

Для прослушивания — кнопку с треугольником.



Всё как на настоящем проигрывателе!

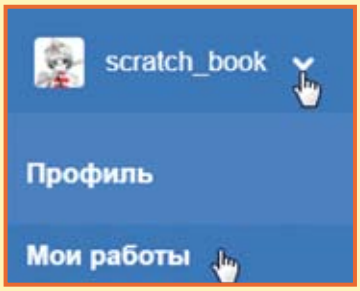
2. С помощью кнопки **Загрузить звук** загрузите в проект свою любимую композицию из файла с расширением mp3. (Надеюсь, вы знаете, что такое расширение файла?)



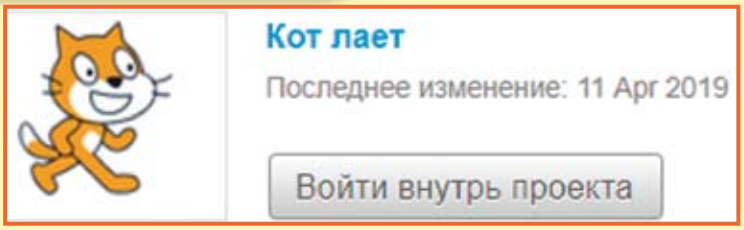
ГЛАВА 2. УСЛОЖНЕНИЕ ПЕРВОГО ПРОЕКТА

2.1. Загрузка первого проекта

Запустите Scratch и войдите в свой аккаунт. Раскройте меню и перейдите в раздел **Мои работы**.

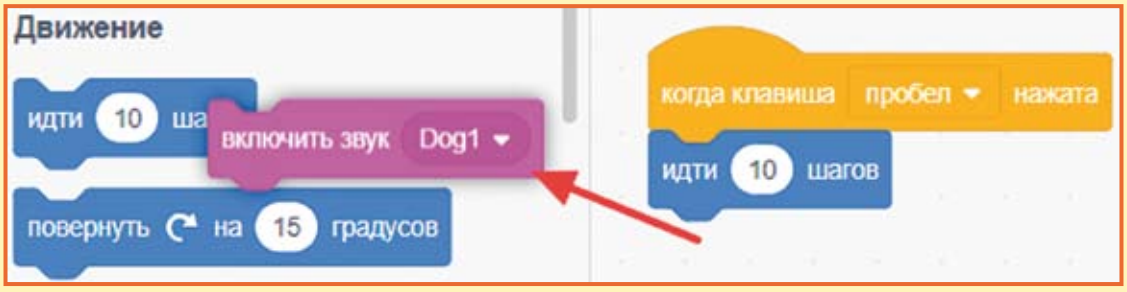


Нажмите кнопку **Войти внутрь проекта**.

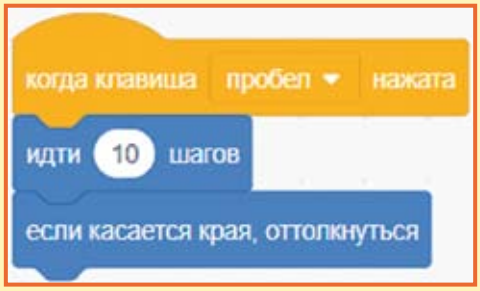


2.2. Изменение скорости движения

Первым делом удалите блок звука, чтобы лай не отвлекал вас от знакомства с новыми блоками. Для этого нажмите на сиреневый блок звука и тащите его обратно в палитру блоков. Отпустите левую кнопку мыши. Когда блок будет находиться над палитрой, он исчезнет.

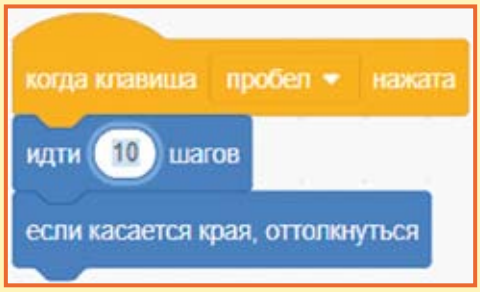


Теперь добавьте к скрипту новый блок из набора блоков **Движение** — если касается края, оттолкнуться.

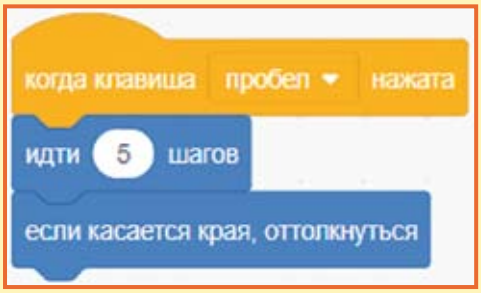


Нажимайте клавишу <Пробел> и посмотрите, что произойдёт, когда Кот дойдёт до края экрана. Он оттолкнётся и пойдёт в обратном направлении! Теперь вам не придётся вытаскивать его за хвост.

А теперь попробуйте изменить скорость движения Кота. Для этого надо изменить значение в блоке идти. Щёлкните на числе **10**. Оно посинеет.

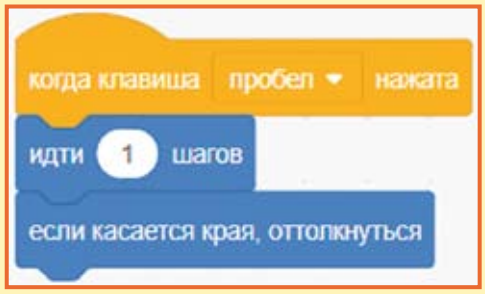


Введите значение 5 и нажмите на клавишу <Enter> (либо щёлкните мышью где-нибудь в пустом месте области скриптов).



Снова нажимайте клавишу <Пробел> — Кот идёт в два раза медленнее! Конечно, ведь пять в два раза меньше десяти.

А теперь введите туда значение 1.

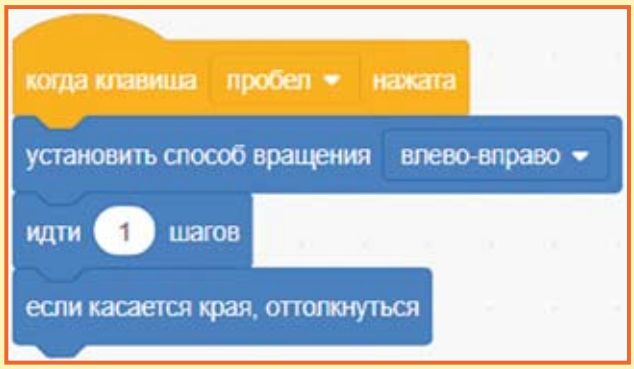


Теперь Кот идёт очень медленно. А что будет, если ввести в блок идти значение больше 10? Поэкспериментируйте.

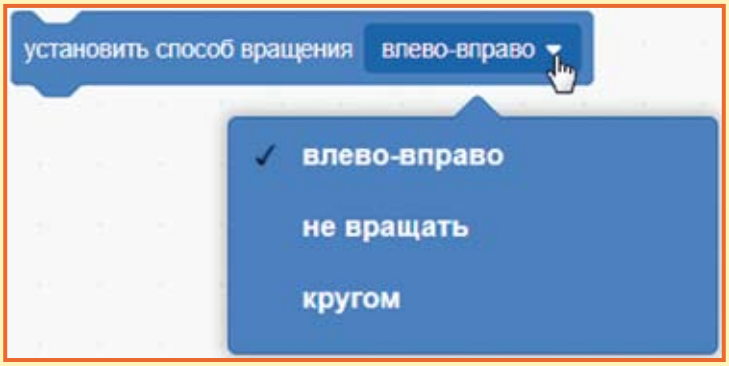
Как вы уже заметили, при движении влево Кот всегда переворачивается вверх тормашками. Это не похоже на движение настоящих котов, обычно они всегда передвигаются тормашками вниз.

Существует блок, который запрещает спрайту переворачиваться. Это установить способ вращения влево-вправо. Добавьте этот блок к скрипту.

Теперь Котик будет бегать лапами вниз. Проверьте, так ли это.



В Scratch существует всего три стиля вращения: «влево-вправо», «не вращать» и «кругом».



У каждого нового спрайта, добавленного в проект, установлен стиль вращения «кругом», поэтому блок стиль вращения мы будем использовать очень часто.

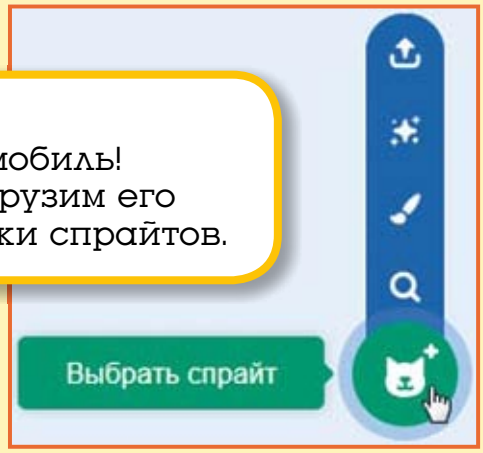
Не забудьте сохранить проект под новым именем: в меню **Файл** выберите команду **Сохранить как копию**.

2.3. Автомобиль с пятью скоростями

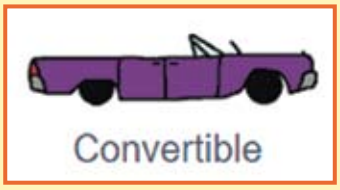
Вы научились перемещать спрайты с различной скоростью, а теперь давайте создадим новый проект, в котором автомобиль, как в жизни, будет иметь пять скоростей.

Создайте новый проект, в котором, как обычно, есть только Кот.

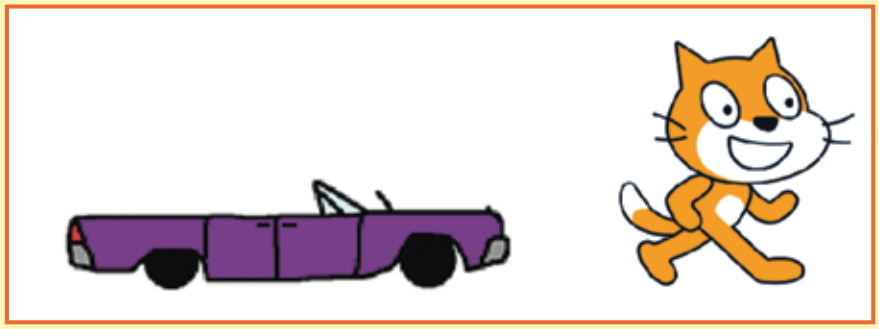
Но нам же нужен автомобиль! Давайте загрузим его из библиотеки спрайтов.



Библиотека спрайтов очень похожа на библиотеку звуков. Выберите понравившийся вам автомобиль, например, вот такой.



Теперь на сцене два спрайта.



Они оба показаны в области спрайтов.

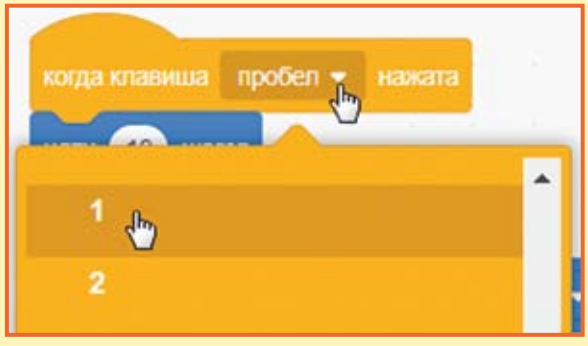
Этот проект будет про автомобиль, поэтому Кот не нужен. Выберите его спрайт и удалите, нажав на крестик.

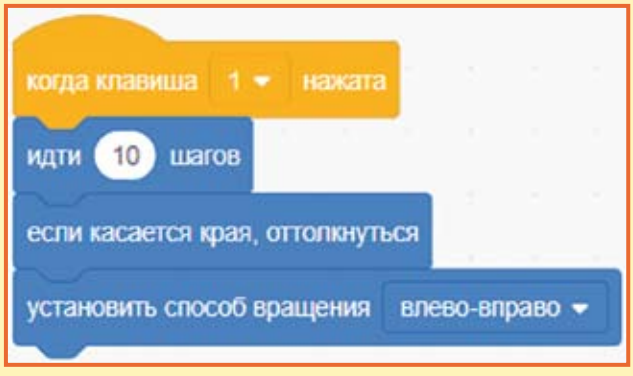
Остался только один спрайт — автомобиль. Теперь его надо запрограммировать. У автомобиля будет 5 скоростей, поэтому нужно создать 5 разных скриптов.

Сначала запрограммируем *первую скорость*. Постройте вот такой скрипт.

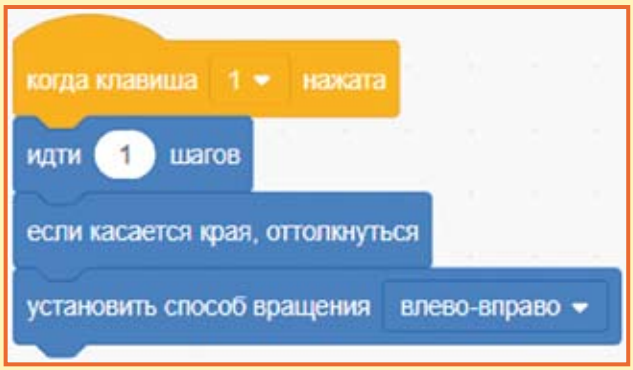
Теперь переключим управление с клавиши <Пробел> на клавишу <1>. Для этого раскройте выпадающий список, нажав на белый треугольничек в блоке когда клавиша пробел нажата, и двигайте курсор вниз.

Выберите единичку.





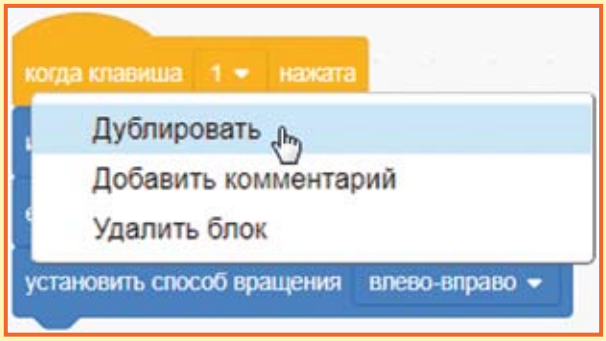
Теперь измените значение в блоке идти на 1. На первой скорости машина будет ехать очень медленно.



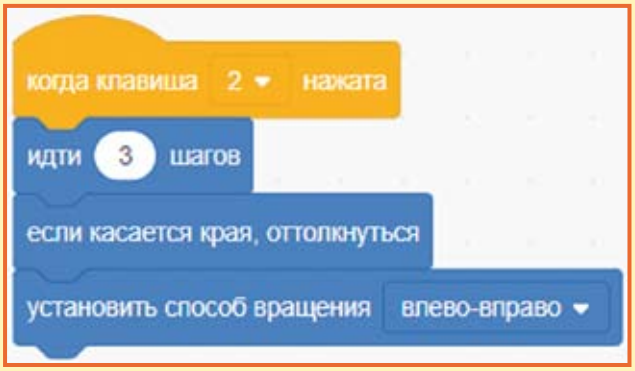
Первый скрипт автомобиля готов! Протестируйте его. Нам надо сделать ещё четыре похожих скрипта.

Совет

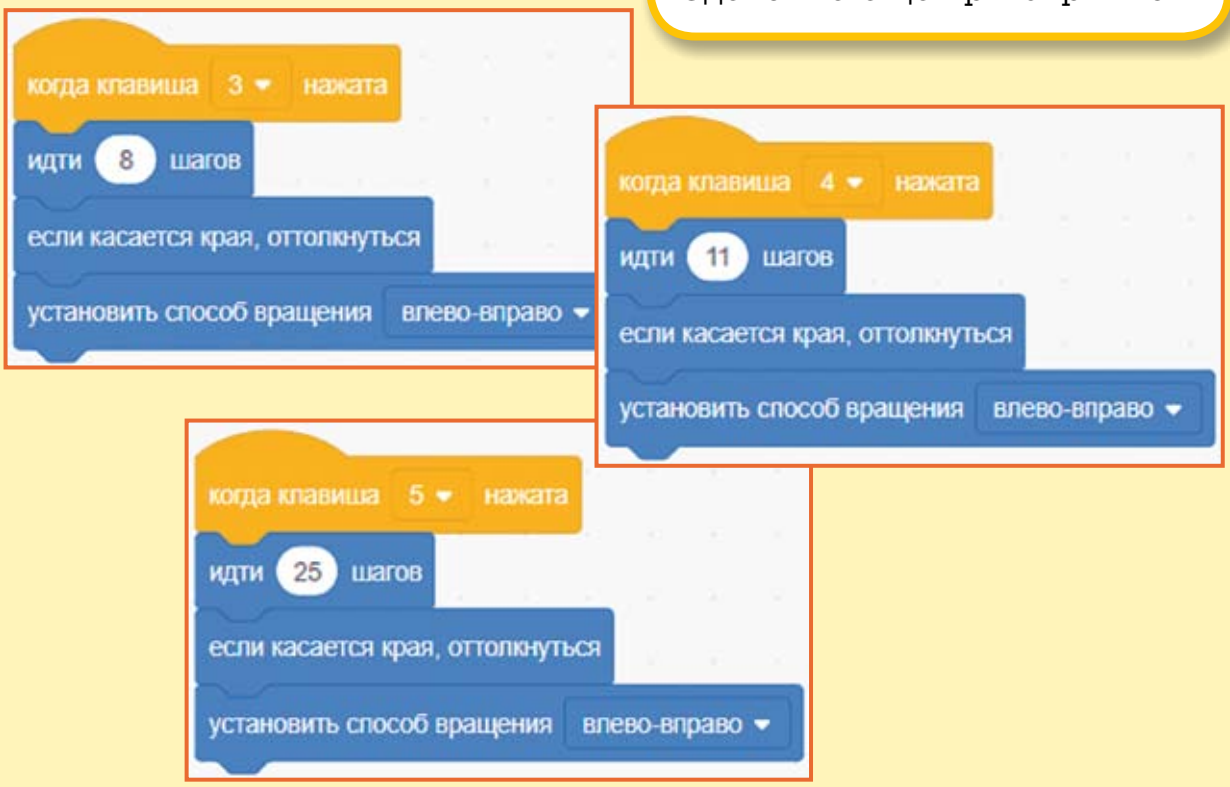
Есть простой способ скопировать скрипт. Щёлкните на верхнем блоке скрипта правой кнопкой мыши и выберите команду **Дублировать.**



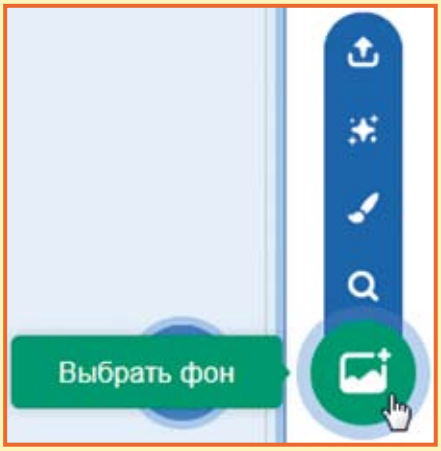
Дублируйте первый скрипт и измените клавишу управления на <2>, а скорость перемещения на 3.

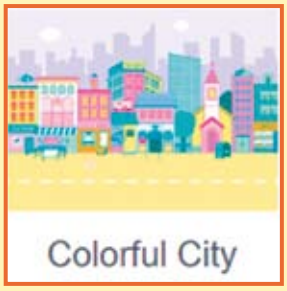


Сделайте ещё три скрипта.



Проект почти готов, давайте украсим сцену — добавим красивый фон. Нажмите кнопку **Выбрать фон**.

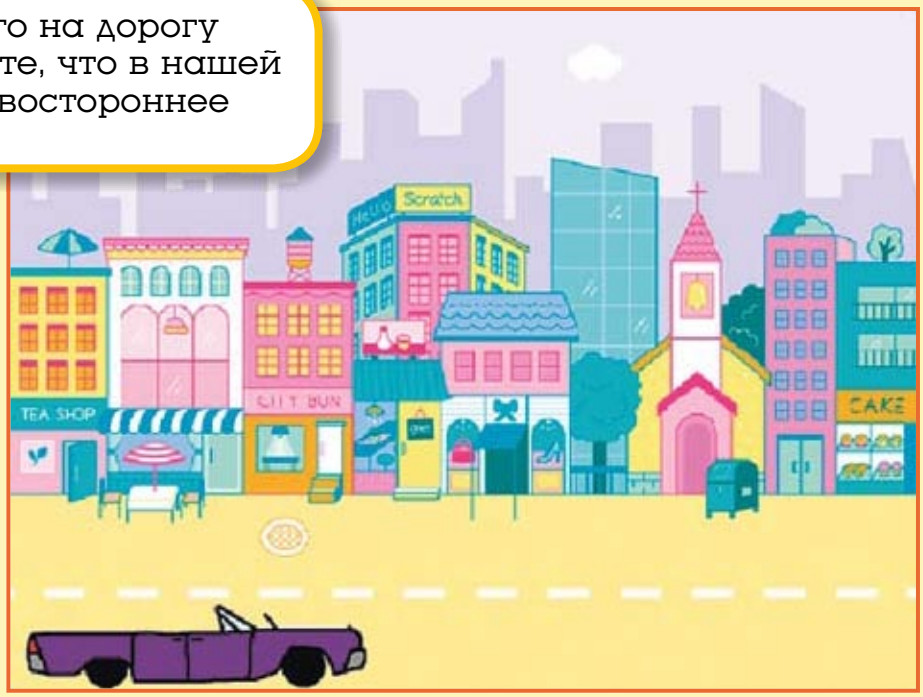




Выберите фон, например, вот этот.

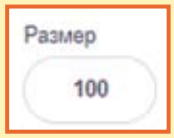
Автомобиль расположен не очень удачно — он стоит на крыше.

Опустите его на дорогу и не забудьте, что в нашей стране правостороннее движение.



Теперь проект готов. Протестируйте его и не забудьте сохранить.

2.4. Задания



1. Добавьте автомобилю шестую скорость.

2. Немного уменьшите размер автомобиля, уменьшая значение в окошке **Размер**.

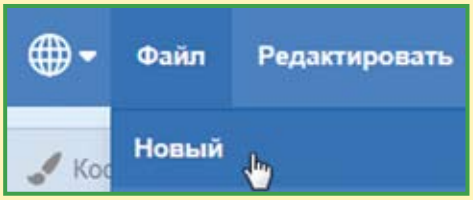
3. Добавьте в проект пешеходов, которые будут стоять на тротуаре. Измените их размер.

4. Добавьте ещё один автомобиль, у которого будет только четыре скорости.

ГЛАВА 3.
ЗНАКОМСТВО С ЭФФЕКТАМИ

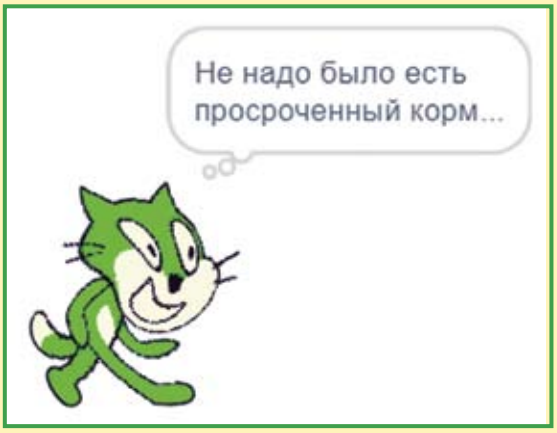
3.1. Создание нового проекта

Создайте новый проект: в меню **Файл** выберите команду **Новый**.



Появится новый проект, в котором, как обычно, есть только Кот. Он-то нам и нужен. Сегодня ему предстоит стать очень эффектным, например таким.

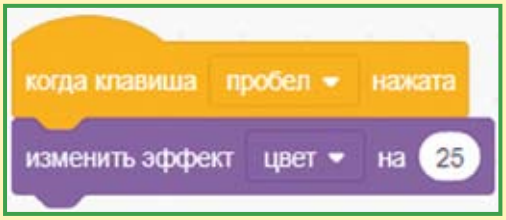
Для изменения внешнего вида Кота мы будем использовать фиолетовые блоки **Внешний вид**.



3.2. Цветовой эффект

Цветовой эффект изменяет цвет спрайта.

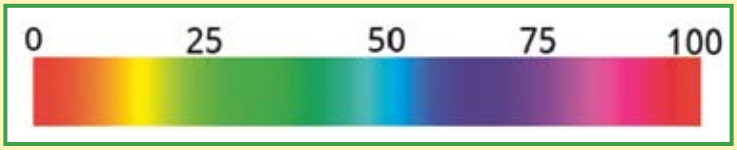
Создайте вот такой скрипт.



Протестируйте его.

При нажатии клавиши <Пробел> Кот меняет цвет. Выглядит весело, но почему так происходит? Дело в том, что все цвета

в Scratch имеют свои числовые значения от 0 до 100. Например, красный цвет имеет значение 0, жёлтый — 16, и так далее (как на картинке).

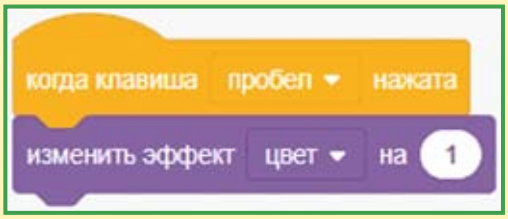


Значение эффекта «цвет» может изменяться в диапазоне от 0 до 200.

И если при создании проекта наш котик был оранжевого цвета № 11, то после нажатия клавиши <Пробел> он станет зелёным.

Таким образом, изменяя эффект «цвет» на 25, Котик пройдёт через испытание палитрой и снова станет рыжим. Это произойдёт ровно через восемь нажатий клавиши <Пробел> ($25 \cdot 8 = 200$). Проверьте это.

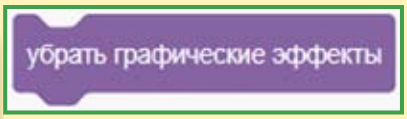
Измените значение 25 на 1. Нажимайте клавишу <Пробел> — теперь цвет изменяется очень плавно.



Вопрос

Сколько раз надо нажать клавишу <Пробел>, чтобы окрас Котика вернулся к рыжему цвету?

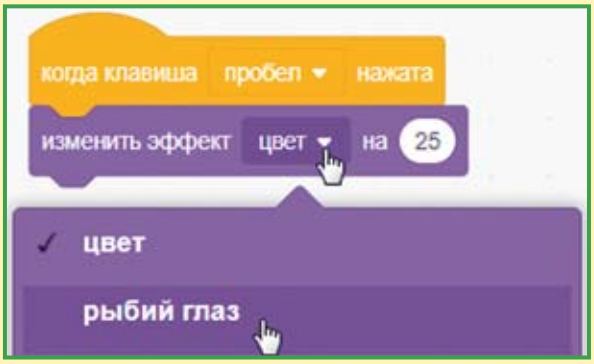
Вернуть Коту первоначальный вид можно, щёлкнув на фиолетовом блоке убрать графические эффекты.



3.3. Эффект рыбьего глаза

Рыбий глаз — очень интересный эффект, раздувающий спрайт.

Выберите этот эффект из выпадающего списка эффектов.
Введите значение 25.



Нажимайте клавишу <Пробел> и смотрите, что получится.

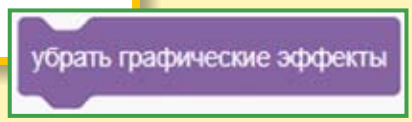
Раздуло Котика не слабо, наверное, вчерашнее молоко было позавчерашним…

В отличие от эффекта цвета, этот эффект бесконечный. Если не отпускать клавишу <Пробел>, то Кот превратится в лепёшку.

Вот таким он станет через 100 нажатий клавиши <Пробел>.

Измените значение 25 на 1. Нажимайте клавишу <Пробел> — теперь Кот раздувается не спеша.

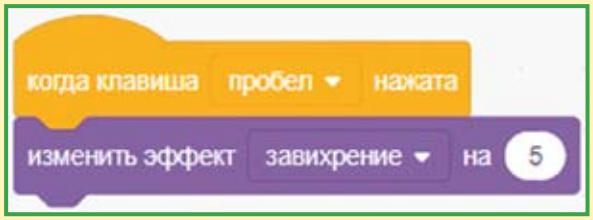
Не забывайте, что вернуть Коту первоначальный вид можно с помощью блока убрать графические эффекты.



3.4. Эффект завихрения

Завихрение — тоже очень зрелищный эффект.

Выберите его из выпадающего списка и введите значение 5.

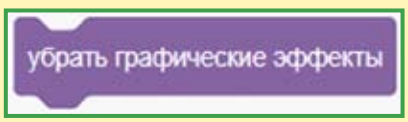


Нажимайте клавишу <Пробел> и посмотрите, как закрутит Котика.

Вот таким он станет через 15 нажатий на <Пробел>, то есть значение эффекта равно 15 · 5 = 75.

Эффект завихрения бесконечный, как и рыбий глаз. Вот каким будет Котик при значении эффекта завихрения в 1000.

Теперь вам точно нужен блок убрать графические эффекты.

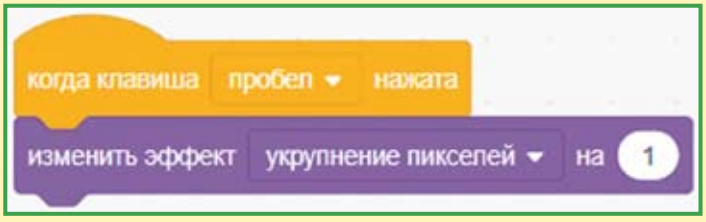


3.5. Эффект укрупнения пикселов

Этот эффект укрупняет точки, из которых состоит изображение — пикселы.

Выглядит это вот так.

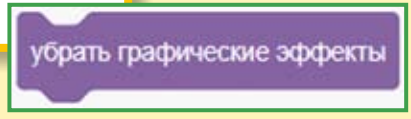
Соберите вот такой скрипт и протестируйте его работу.



Посмотрите, что будет при значении эффекта 100 и больше.

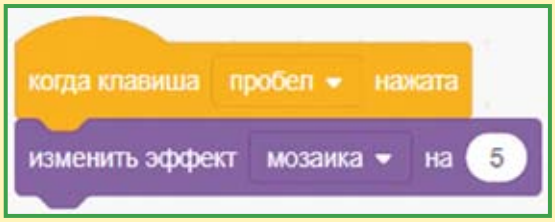
При значении эффекта 400 Кот превратится в серый кирпич, а при значении больше 700 совсем исчезнет.

Не забывайте о блоке убрать графические эффекты!



3.6. Эффект мозаики

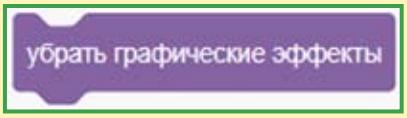
Этот эффект работает вот так. Спрайт превращается в мозаику.



Соберите вот такой скрипт и протестируйте его.

При значении 100 и более спрайт выглядит как армия малюсеньких клонов, но это по-прежнему один спрайт.

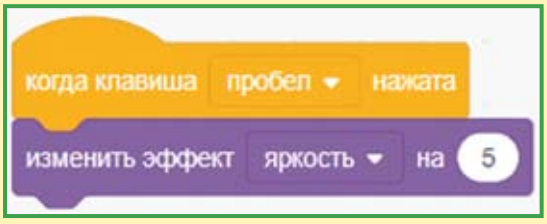
Верните Котику первоначальный вид.



3.7. Эффект яркости

Этот эффект увеличивает яркость спрайта.

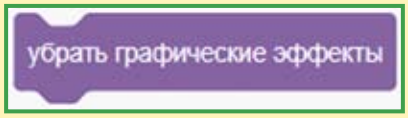
Сделайте вот такой скрипт и посмотрите, как он работает.



Вот как будет выглядеть Котик после десяти нажатий клавиши <Пробел>.

При значении эффекта 100 Кот исчезнет. Таким образом, значение эффекта яркости может изменяться от 0 до 100.

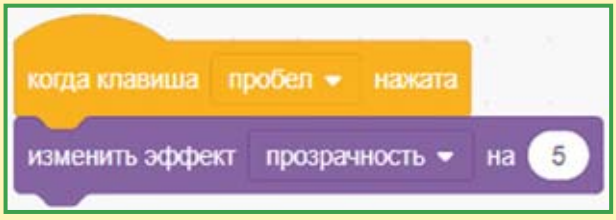
Не забывайте о блоке убрать графические эффекты!



3.8. Эффект прозрачности

Этот эффект позволяет спрайту исчезать и появляться. Он изменяет прозрачность спрайта.

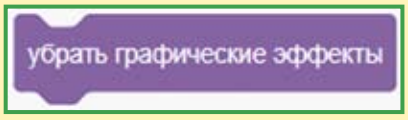
Протестируйте его работу.



Вот так будет выглядеть Котик при значении эффекта прозрачности равном 50.

При значении эффекта прозрачности равном 100 спрайт станет невидимым.

Верните Котику первоначальный вид.



3.9. Анимация

Для того чтобы при движении Кот не плыл, а шевелил лапами, надо применить блок следующий костюм.

Этот блок переключает по очереди все костюмы спрайта.

Нажимайте клавишу <Пробел>, и Кот зашагает! Это происходит из-за того, что у Кота два костюма.

Перейдите на вкладку **Костюмы**.

Здесь вы можете увидеть, что у спрайта два костюма с именами «костюм 1» и «костюм 2».

Нажимайте клавишу <Пробел>, и вы увидите, как костюмы по очереди переключаются.

3.10. Вопросы

1. Какие эффекты применены к Коту?

2. А теперь?

3. А здесь что с Котом?

4. А здесь какие эффекты?

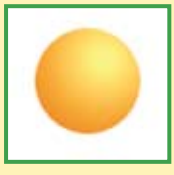
3.11. Задания

1. Добавьте в новый проект спрайт чёрного щенка и примените к нему поочерёдно все эффекты.

Некоторые эффекты работать не будут. Подумайте, почему?

2. А теперь добавьте в проект шарик.

Действие каких эффектов будет почти незаметно?

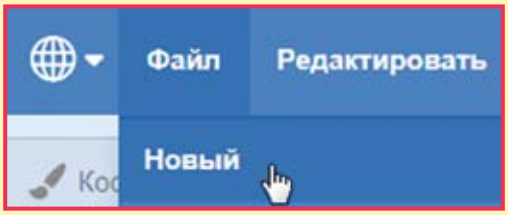


3. Добавьте в проект фон сцены и потренируйтесь на нём. Вот, например, как действует на фон эффект завихрения.

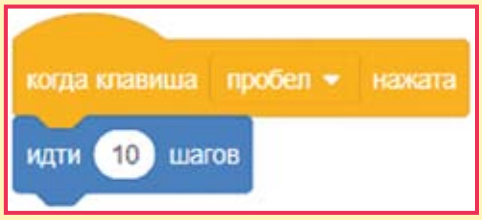
ГЛАВА 4. ЗНАКОМСТВО С ОТРИЦАТЕЛЬНЫМИ ЧИСЛАМИ

4.1. Ходим задом наперёд

Создайте новый проект: в меню **Файл** выберите команду **Новый**.



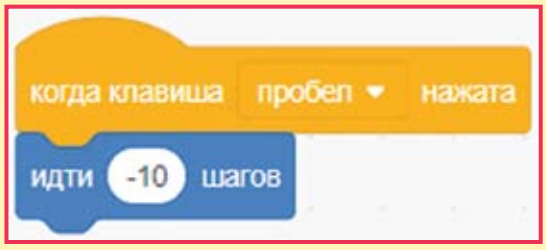
Появится новый проект, в котором, как обычно, есть только Кот. Сделайте простой скрипт гуляющего Кота.



Нажимайте клавишу <Пробел>. Кот, как обычно, гуляет по сцене.

Для того чтобы научить его ходить задом наперёд и делать всё наоборот, нам понадобится отрицательное число −10. Это то же самое, что и 10, только с минусом. Отрицательные числа очень похожи на обыкновенные числа, только они делают всё наоборот, право меняют на лево, верх на низ, перёд на зад, разгон на торможение, рост на снижение, взлёт на падение и так далее.

Замените в скрипте 10 на −10.

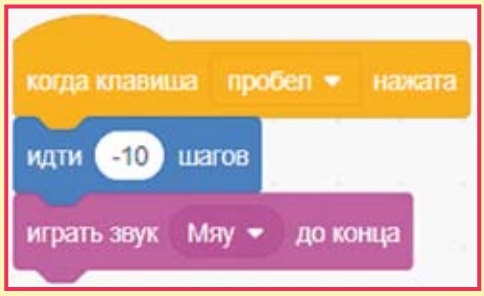


Протестируйте скрипт, Кот идёт задом наперёд!

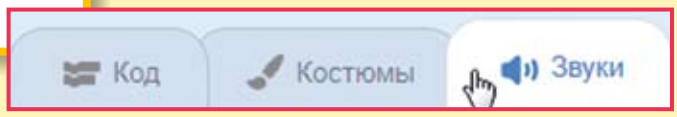
4.2. Переворачиваем звуки

А теперь давайте научим его мяукать задом наперёд! Это можно сделать и без отрицательных чисел.

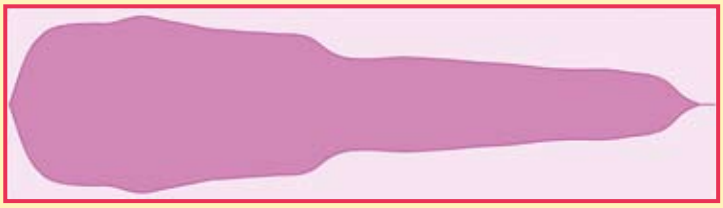
Добавьте к скрипту блок мяуканья.



Перейдите на вкладку **Звуки**.



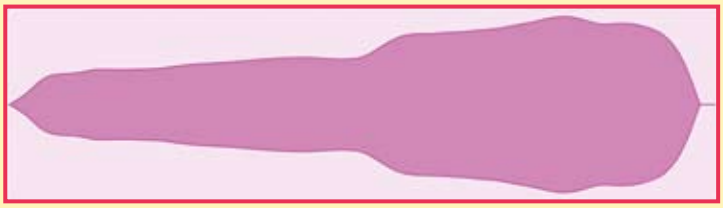
Здесь вы увидите графическое изображение звука. Чем шире полоса, тем звук громче.



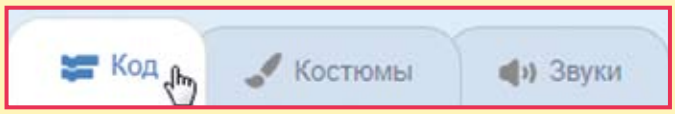
Нажмите на кнопку с треугольником.

Вы услышите звук. Для того чтобы проиграть его задом наперёд, нажмите кнопку **Развернуть**.

Выделенный звук развернётся задом наперёд.



Перейдите на вкладку **Код**.



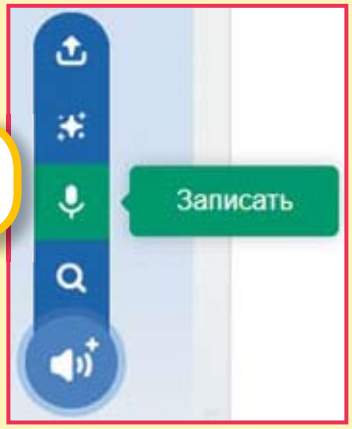
Нажмите клавишу <Пробел> и послушайте, как стал мяукать Кот.

Эксперимент

Добавьте Коту другие звуки и переверните их. Какой самый прикольный? Мне понравился перевёрнутый звук triumph.

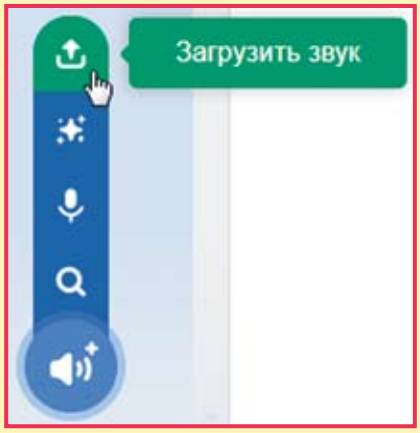
Перевернуть задом наперёд можно не только звуки из библиотеки, это можно сделать и с записанным звуком.

Нажмите кнопку **Записать**.



Затем нажмите оранжевую кнопку **Записать**.

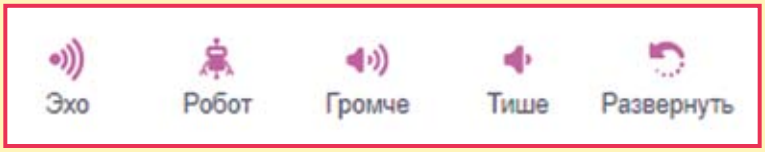
Записанный звук также можно перевернуть, таким способом можно получить очень интересные звуки для ваших проектов.



Также можно переворачивать звуки, скачанные из Интернета.

Для того чтобы загрузить звук или музыку из файла, надо нажать кнопку **Загрузить звук** (на вкладке **Звуки**).

В открывшемся окне необходимо выбрать нужный файл и нажать кнопку **Открыть**. Теперь можно проиграть файл задом наперёд или применить к нему другие звуковые эффекты.



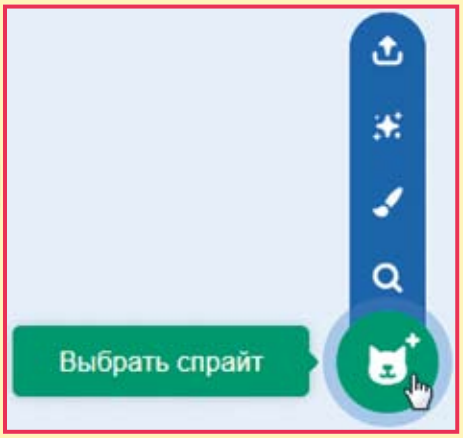
Совет

Загружайте только файлы формата MP3 или WAV! Старайтесь не загружать файлы большого размера. Если в вашем проекте используется только часть музыкального файла, то удалите его неиспользуемую часть, выделив её и нажав клавишу <Delete>.

4.3. Волшебный единорог

Используя отрицательные числа, можно сделать много интересных проектов. Например, про Единорога, появляющегося из ниоткуда.

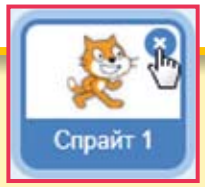
Откройте библиотеку спрайтов.



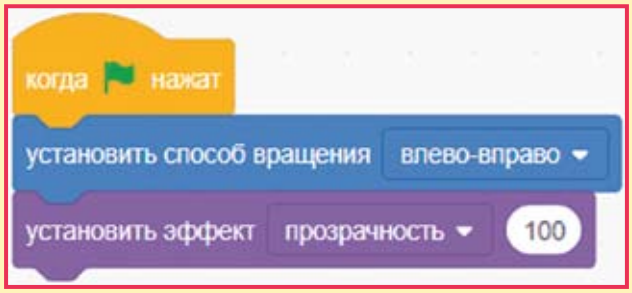
Выберите бегущего единорога.

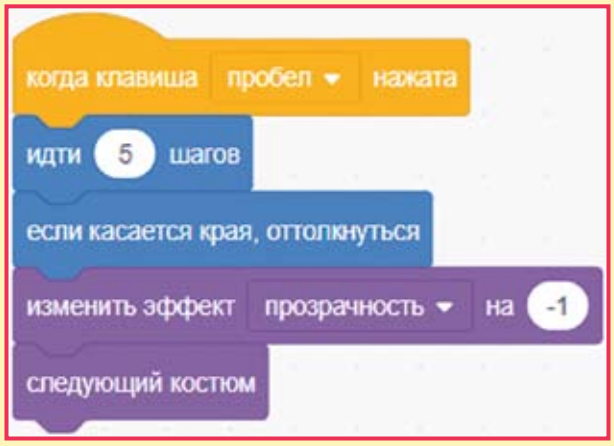
Теперь у нас два спрайта.

Кот нам не нужен, удалите его, нажав на крестик.



Сделайте вот такие два скрипта для Единорога.

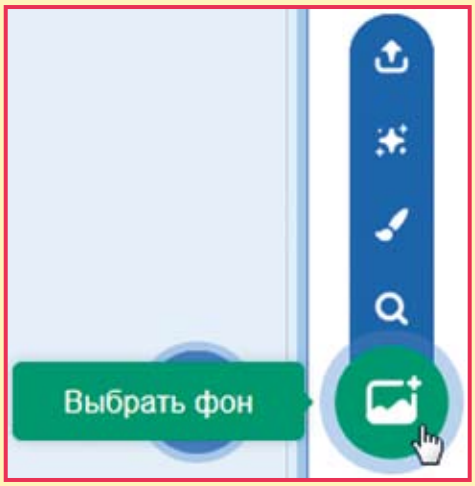




Протестируйте работу скриптов. Щёлкните на зелёном флажке (он находится в правом верхнем углу сцены) — Единорог исчезнет. Нажимайте клавишу <Пробел>, и он появится из ниоткуда! Но почему это происходит? Всё очень просто. Когда вы нажимаете на зелёный флажок, эффект призрака становится

равным 100, и Единорог исчезает. При нажатии клавиши <Пробел> значение эффекта призрака каждый раз уменьшается на 1 и после ста нажатий становится равным нулю.

Давайте добавим красивый фон. Нажмите кнопку **Выбрать фон**.



Снова нажмите на зелёный флажок, а затем на клавишу <Пробел>. Так гораздо красивее!

Сохраните проект.

4.4. Вопросы

1. Что получится, если мы изменим 5 на −5 в блоке идти?

2. Что получится, если в начале проекта мы установим прозрачность Единорога в значение 0, а затем при нажатии клавиши <Пробел> будем изменять её на 1?

3. На улице было +25 градусов, а затем температура изменилась на −5 градусов. Какая температура стала на улице?

4. У Буратино было 5 золотых, а потом Лиса Алиса подарила ему −1 золотой, сколько золотых стало у Буратино?

4.5. Задания

1. С помощью инструмента записи голоса запишите какую-нибудь фразу и переверните её задом наперёд.

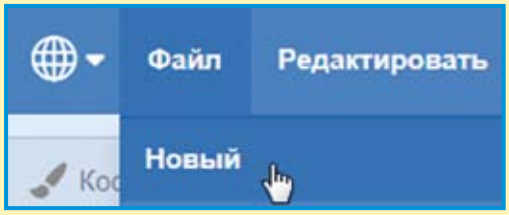
2. Добавьте в проект с Единорогом волшебную музыку из библиотеки звуков.

3. Добавьте в проект с Единорогом ещё одного Единорога, который будет появляться при нажатии на клавишу <↑>.

ГЛАВА 5. ЗНАКОМСТВО С ПЕРОМ

5.1. Рисуем каракули

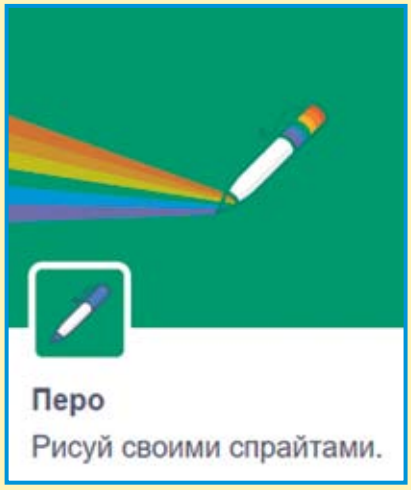
Создайте новый проект: в меню **Файл** выберите команду **Новый**.



Появится новый проект, в котором, как обычно, есть только Кот, и сейчас мы научим его рисовать на сцене. Для этого мы будем использовать зелёные блоки **Перо**. Для работы с ними нужно добавить специальное расширение.

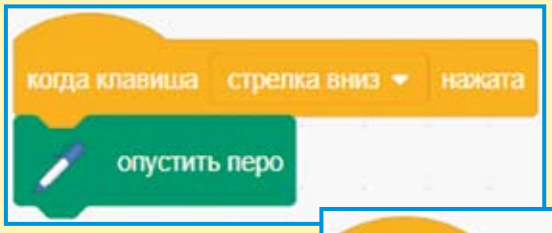
Нажмите на кнопку.

Добавьте расширение **Перо**.

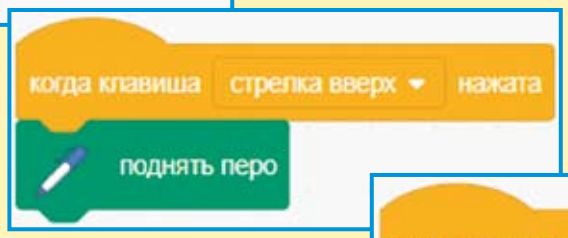


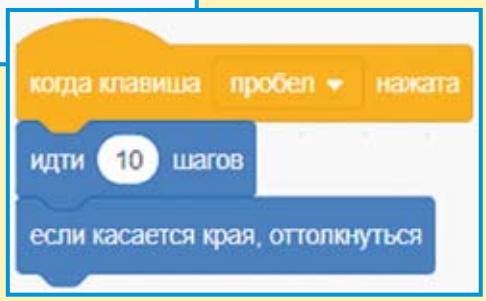
Теперь в проект можно добавлять блоки пера.

Перо — это такой цветной фломастер, которым спрайт может рисовать на сцене. Перо может быть поднято — тогда спрайт при движении ничего не рисует, или опущено — тогда при движении спрайта за ним тянется линия.



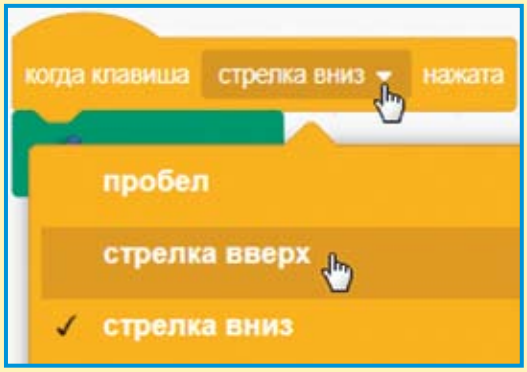
Сделайте вот такую программу для Кота.





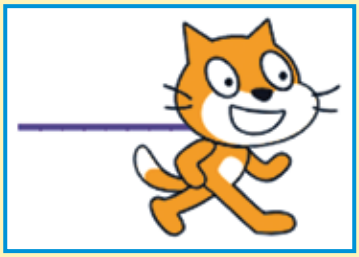
Подсказка

Для выбора нужной клавиши раскройте выпадающий список.

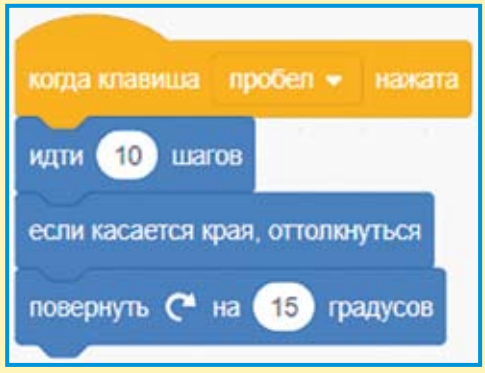


Просто нажимайте клавишу <Пробел> — Котик, как обычно, гуляет по сцене.

А теперь нажмите клавишу <↓>, а затем снова нажимайте <Пробел>. Перо будет опущено, и за Котом потянется тонкая синяя линия.

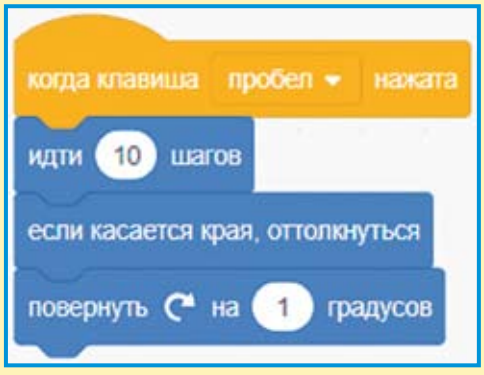


Если потом нажать клавишу <↑>, то перо будет поднято, и линия появляться не станет.

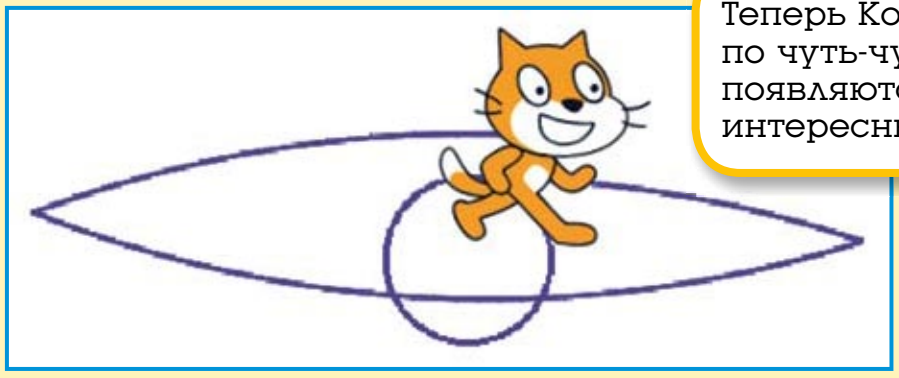


Немного повеселимся, добавьте к скрипту блок поворота.

Нажимайте клавишу <Пробел> — Кот нарисует окружность!

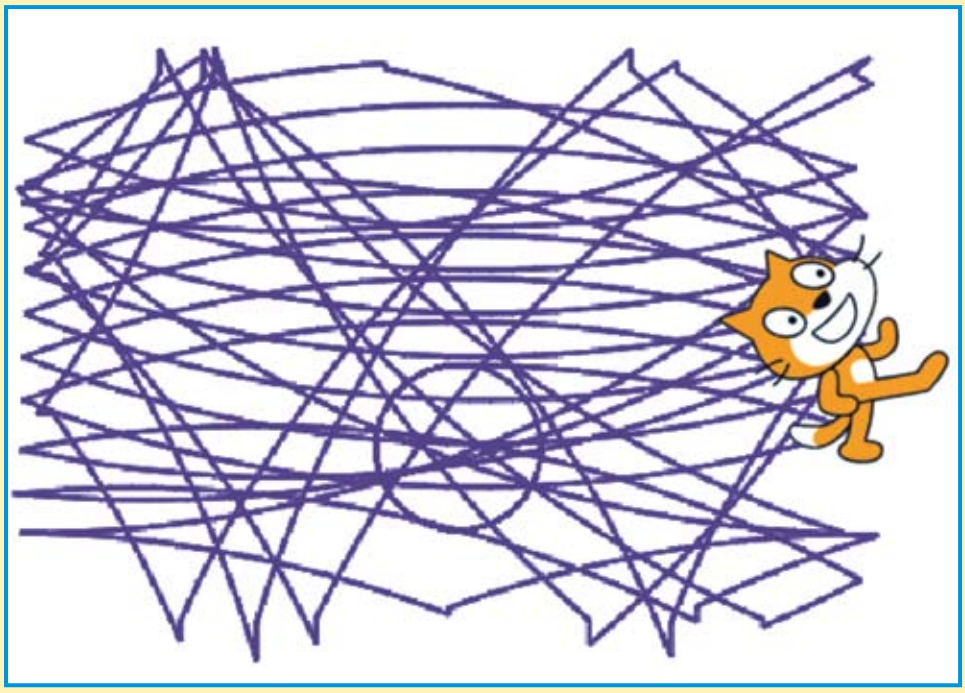


Это происходит потому, что каждый раз Кот делает 10 шагов вперёд и немного поворачивается по часовой стрелке. Задайте угол поворота 1 градус.



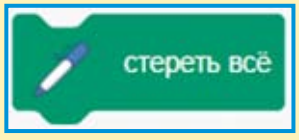
Теперь Кот поворачивает по чуть-чуть, и на сцене появляются очень интересные узоры!

Если нажимать клавишу <Пробел> много раз, то вся сцена будет размалёвана.



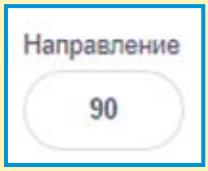
Совет

Для того чтобы очистить сцену, щёлкните мышью на блоке стереть всё.



Совет

Для того чтобы повернуть Кота в исходное положение, так сказать, поставить его на лапы, нужно просто ввести число 90 в окошке **Направление**.

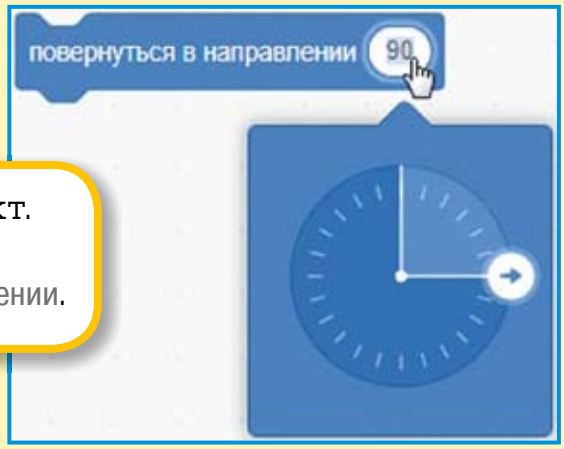


Эксперимент
Изменяйте значения в блоках идти и повернуть.

Да уж... Художник из Котика получился не очень...

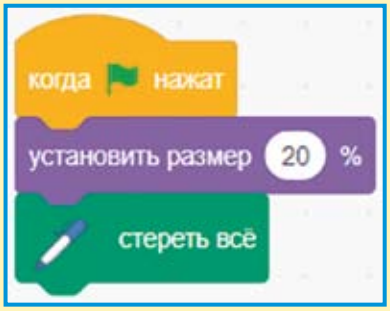
5.2. Рисуем красиво

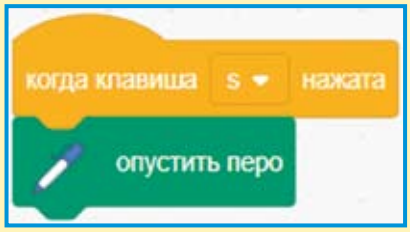
Давайте будем управлять движением Кота с помощью стрелок на клавиатуре. Может, тогда получится нарисовать что-нибудь красивое.

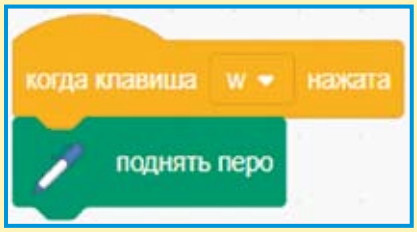


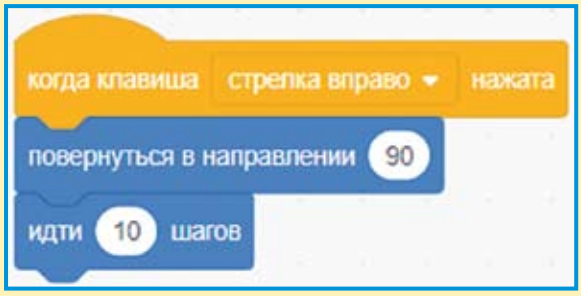
Создайте новый проект. Здесь вам пригодится блок повернуться в направлении.

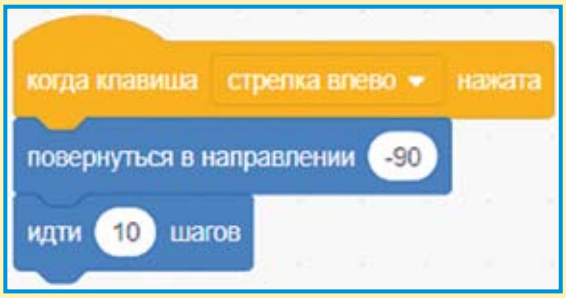
Этот блок поворачивает спрайт в указанном направлении. Соберите для Кота вот такую программу. Она состоит из семи скриптов.

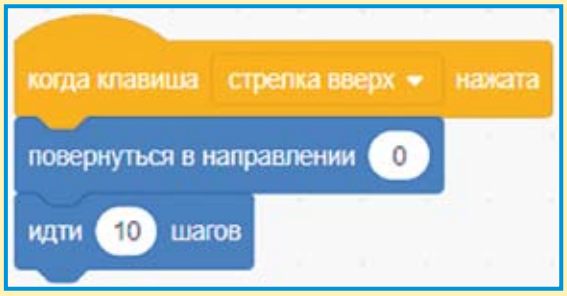


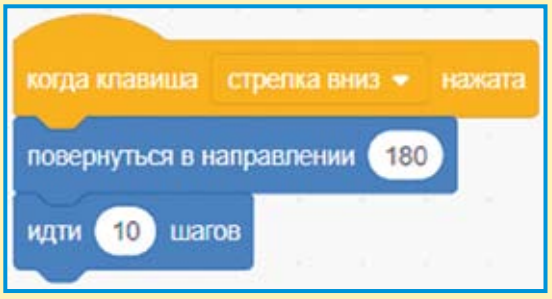












При нажатии на зелёный флажок Котик уменьшится в размере и очистит сцену.

Подсказка

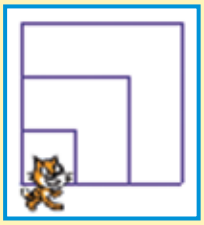
Нормальный размер спрайта равен 100% . Размер 20% ровно в 5 раз меньше нормального, а размер 50% ровно в 2 раза меньше нормального размера.

При нажатии клавиши <s> перо опустится, а при нажатии <w> перо поднимется. При нажатии клавиш-стрелок Котик будет поворачиваться в соответствующем направлении и делать по 10 шагов.

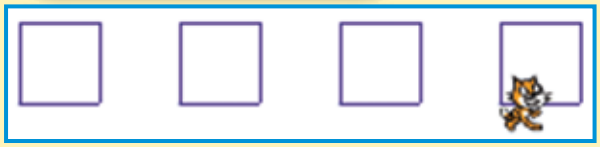
Совет

Если перо не опускается и не поднимается, проверьте, что вы переключились на английскую раскладку клавиатуры.

Теперь можно нарисовать всё,
что состоит из прямоугольников.
Попробуйте нарисовать три квадрата.

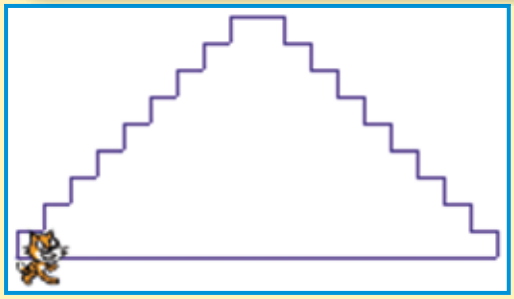


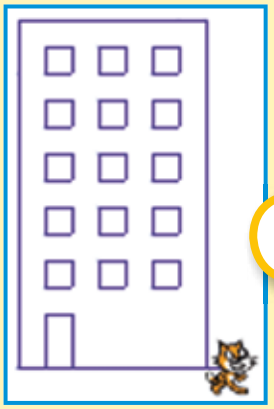
Ряд из квадратов.



Спираль.

Пирамиду индейцев майя.



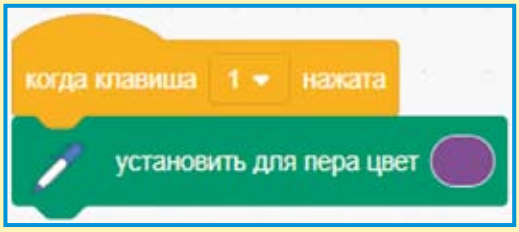


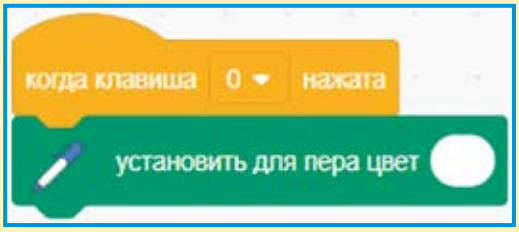
Многоэтажку.

Многоэтажку нарисовать не просто. У меня не получилось с первого раза. Любая ошибка, и приходится начинать сначала.

Давайте добавим в проект возможность стереть нарисованное. А что значит стереть нарисованное на белом фоне? Это значит закрасить белым! Нужно добавить в проект возможность рисовать белым.

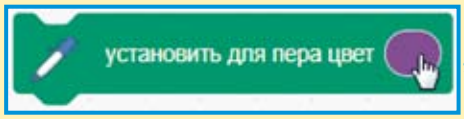
Добавьте в проект два скрипта.





Подсказка

Для выбора цвета в блоке установки цвета необходимо щёлкнуть в цветной овальчик



, выбрать пипетку  , а затем щёлкнуть в любом месте на сцене, окрашенном в нужный цвет.

Теперь при нажатии клавиши с цифрой 0 Котик будет стирать нарисованное, а при нажатии на клавишу с 1 снова сможет рисовать.

Сохраните проект.

5.3. Вопросы

1. Кот рисует короткими отрезками. Как увеличить длину отрезков?

2. Как изменить программу, чтобы она стирала нарисованное на сером фоне?

3. Что будет с Котом, если установить его размер не 20%, а 200%?

5.4. Задания

1. Добавьте в проект возможность рисовать красным цветом.

2. Установите размер кота в 10 раз меньше нормального.

3. Сделайте так, чтобы можно было очищать сцену, используя клавиатуру, не нажимая на зелёный флажок.

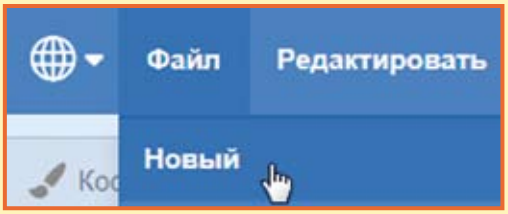
4. Для того чтобы было удобнее рисовать, сделайте так, чтобы цвет Кота был разным при опущенном пере и при поднятом пере.

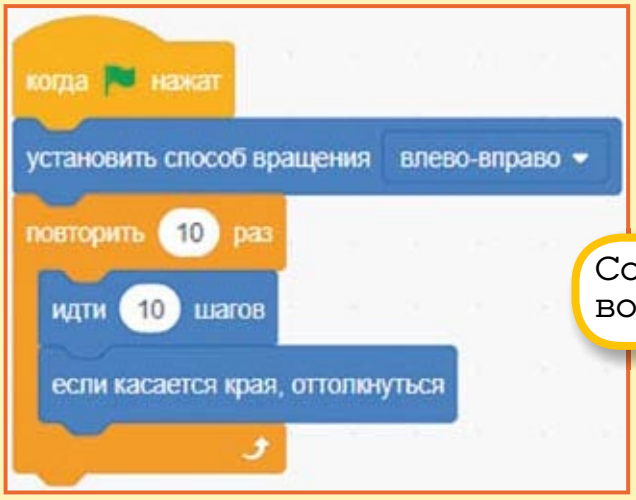
ГЛАВА 6. ЦИКЛЫ

6.1. Знакомство с циклами

В предыдущих главах для управления спрайтами вам приходилось постоянно нажимать клавиши. Это не всегда удобно. Для автоматического выполнения действий в Scratch есть очень важный оранжевый блок повторить. Он позволяет повторить одно действие несколько раз подряд. Давайте попробуем.

Создайте новый проект: в меню **Файл** выберите команду **Новый**.





Соберите для Кота вот такой скрипт.

Блок повторить выполнит своё содержимое 10 раз.

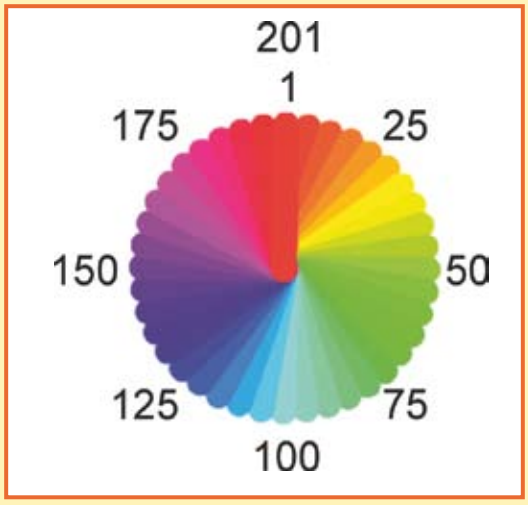
Щёлкните по зелёному флажку. Кот смело пошагает вперёд и сделает 10 раз по 10 шагов — всего 100 шагов. Снова щёлкните по флажку — Кот опять сделает 100 шагов.

Эксперимент

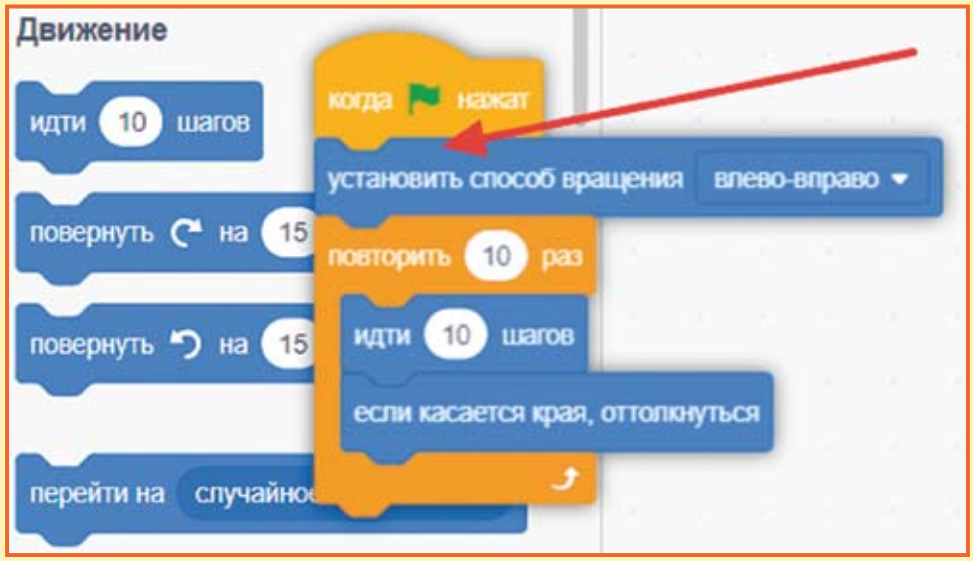
Вводите разные значения в блок идти и в блок повторить и посмотрите, что получится. Не вводите в блок повторить значения больше 1000.

6.2. Циклы и эффект цвета

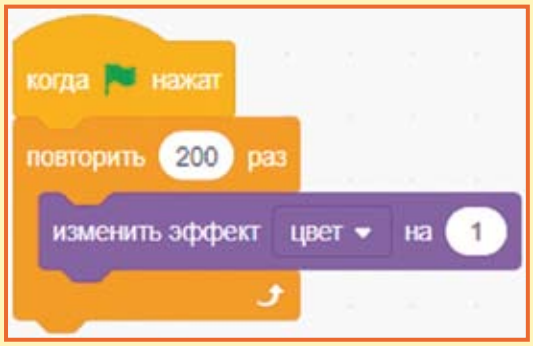
Вы уже знаете, что эффект «цвет» может изменяться в диапазоне от 0 до 200. Вот как будет изменяться цвет красного спрайта при использовании этого эффекта.



Давайте сделаем Кота, который будет автоматически изменять свой цвет 200 раз и снова становиться рыжим. Удалите созданный скрипт: для этого перетащите его целиком в палитру блоков и отпустите кнопку мыши. Скрипт исчезнет.



Теперь создайте для Кота вот такой скрипт.



Нажмите на зелёный флажок. Кот будет плавно изменять свой цвет, пока снова не станет рыжим.

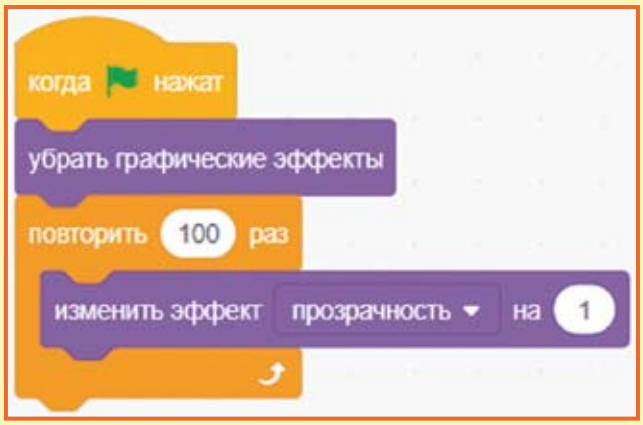
Эксперимент

Измените значение в блоке повторить на 400. А сколько раз теперь Кот пройдёт по цветовому кругу? Измените значение в блоке изменить цвет эффект на 2. Что изменится в работе скрипта?

6.3. Циклы и эффект прозрачности

А теперь давайте поиграем с эффектом прозрачности. Этот эффект изменяется от 0 до 100.

Создайте для Кота вот такой скрипт.



При нажатии на зелёный флажок сначала пропадут все графические эффекты, а затем Кот плавно исчезнет.

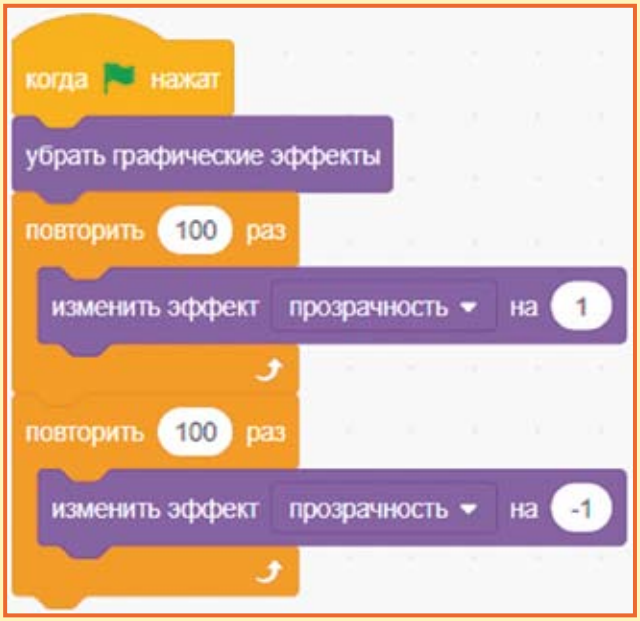
Эксперимент

Вводить значение больше 100 в блок повторить не имеет смысла, ведь Кот не может стать дважды невидим!

А вот значение в блоке изменить эффект прозрачность поменять можно. На что это повлияет?

Очень жаль, что Котик постоянно пропадает. Давайте сделаем так, чтобы он плавно исчезал, а потом так же плавно появлялся. Для этого нам понадобится отрицательное число −1. Это число имеет перед собой знак «минус», поэтому, изменяя эффект призрака на −1, мы будем уменьшать его значение от 100 до 0, пока Котик не станет таким как был.

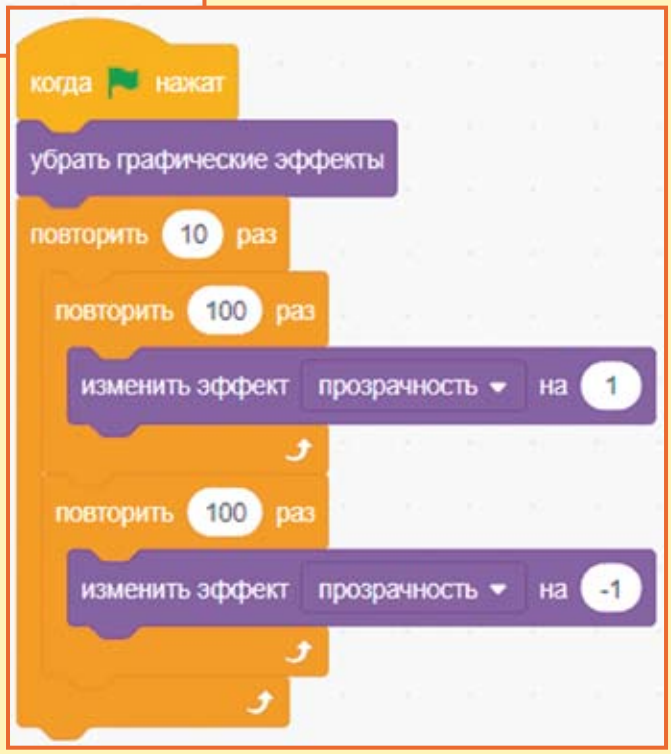
Доработайте скрипт.



Проверьте, как работает этот скрипт. Нажмите на зелёный флажок. Кот уйдёт в страну призраков, а потом вернётся из неё!

А что, если нам надо запустить Кота в страну призраков 10 раз подряд? Для этого может пригодиться ещё один блок повтора! Выберите его в палитре блоков и подтащите к скрипту так, чтобы он обнял сразу два блока повтора.

Получится вот такой скрипт.



Запустите его. Кот станет невидимым 10 раз подряд. Сохраните проект.

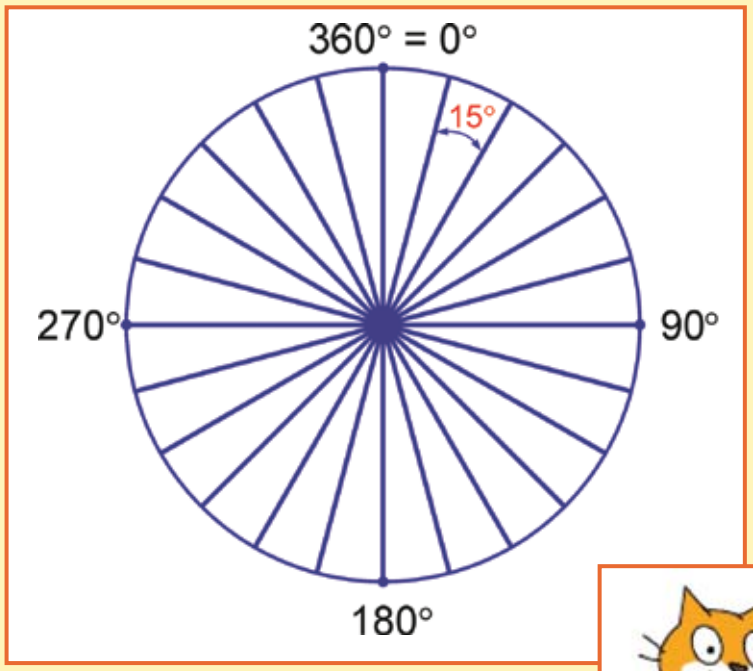
6.4. Вращение

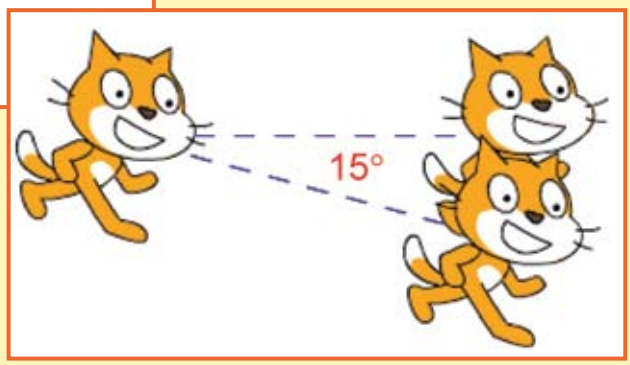
А теперь давайте познакомимся с блоками поворота. Их всего два: поворот по часовой стрелке и поворот против часовой стрелки.

Как вы, наверное, уже знаете, окружность делится на 360 градусов, поэтому спрайт в Scratch может двигаться в одном из 360 направлений.

Градусы обозначаются маленьким кружочком, например, 360°.

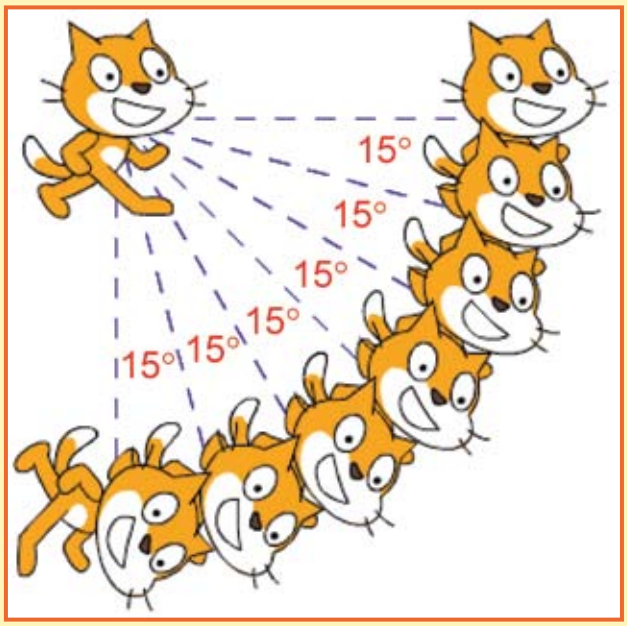
Движение вверх — это движение в направлении 0°, движение вправо — движение в направлении 90° и так далее.

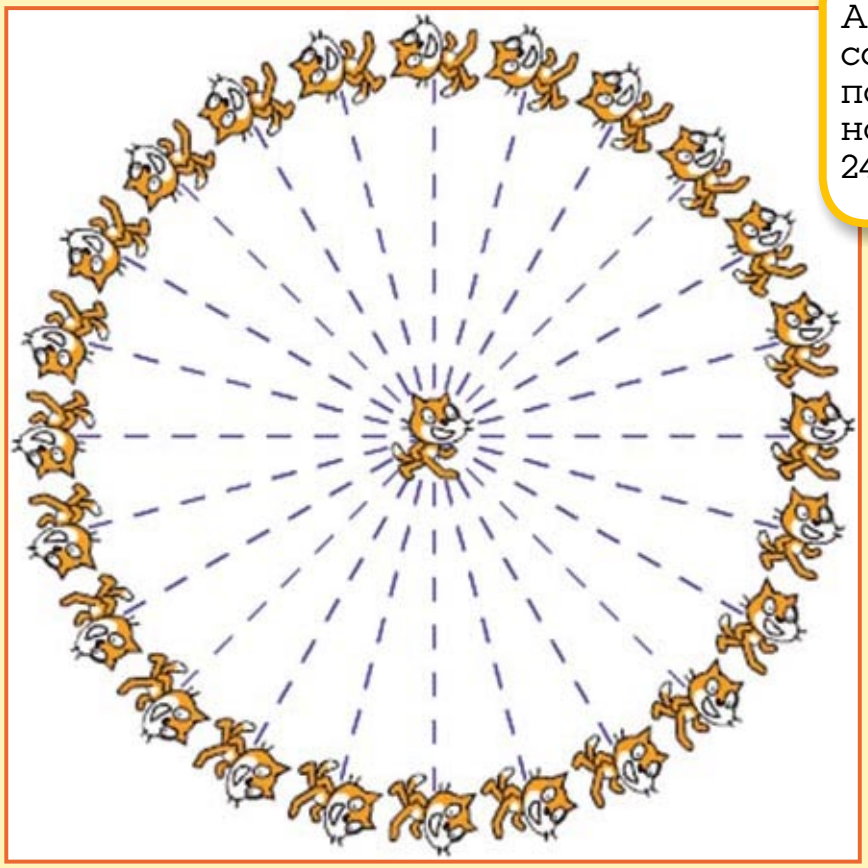




Поворот на 15° — это совсем немного.

Для того чтобы повернуть персонаж под углом в 90°, надо 6 раз повернуть на 15°.

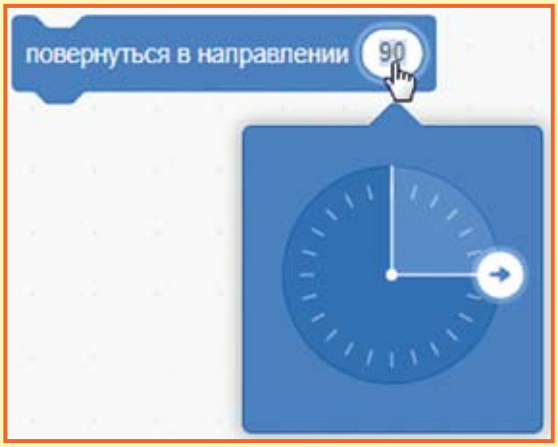




А для того чтобы совершить полный оборот, надо повернуться 24 раза по 15°.

Совет

В блоке повернуться в направлении 90 спрятана подсказка по направлениям. Если вы забудете, где право и где лево, то всегда сможете подсмотреть правильный ответ.



Обратите внимание, что направления −90° и 270° — это одно и то же.

Для того чтобы лучше понять, что такое градусы, создайте новый проект и сделайте такой скрипт.

Вы уже знаете, что 24 раза по 15° — это полный оборот. Нажмите на зелёный флажок и убедитесь в этом. Кот сделает ровно один оборот!

Эксперимент

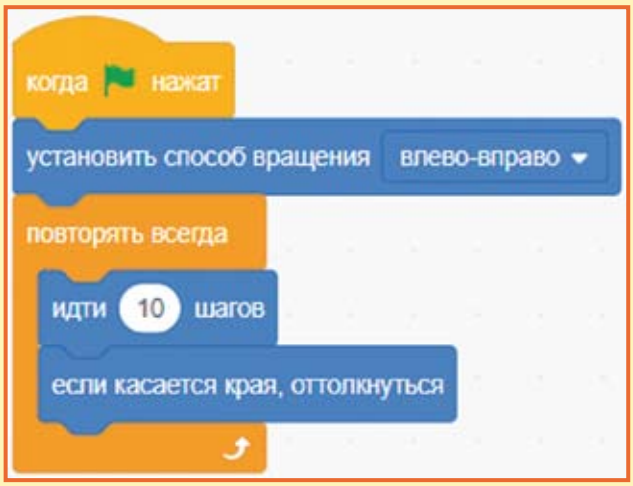
Изменяйте значения в блоках повторить и повернуть так, чтобы произведение этих чисел было равно 360. Например, 36 и 10, 72 и 5, 10 и 36, 5 и 72, 2 и 180 и так далее. Обратите внимание на скорость вращения Кота.

6.5. Бесконечный цикл

Интересно, а что будет, если ввести в блок повторить огромное значение, например, один миллион — 1 000 000? В этом случае прогулка Кота затянется на 8 часов, а если ввести один миллиард — 1 000 000 000, то Котик будет гулять целый год!

Однако для бесконечного выполнения скрипта лучше использовать блок повторять всегда. Этот блок всегда повторяет своё содержимое до бесконечности.

Создайте для Кота вот такой скрипт.



Щёлкните по зелёному флажку. Теперь Кота ничто не остановит! Можете съездить на лето к бабушке, а вернувшись, убедиться, что Кот всё ещё ходит по сцене кругом. «Идёт направо — песнь заводит, налево — сказку говорит».

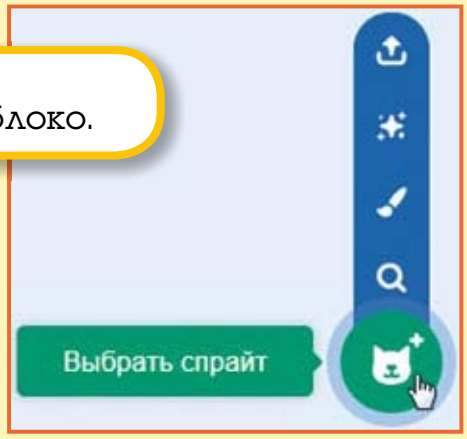
Шутка. Кота-сказочника вы сможете запрограммировать после того, как познакомитесь с условным блоком.

6.6. Автоматическая печать

Давайте сделаем небольшой проект с применением блока повторять всегда. Украсим двор яблоками!

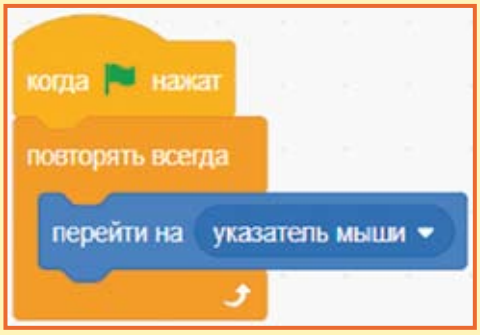
Создайте новый проект.

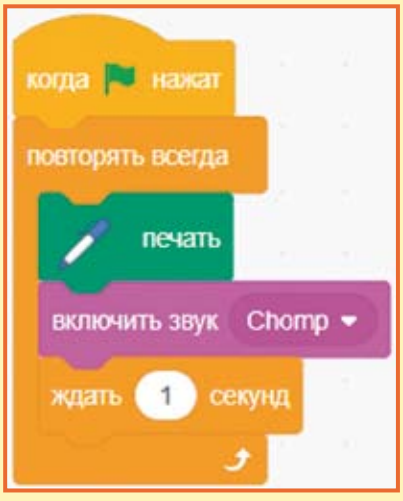
Войдите в библиотеку спрайтов и выберите Яблоко.



Кота удалите знакомым вам способом.

Сделайте вот такую программу из двух скриптов для яблока.





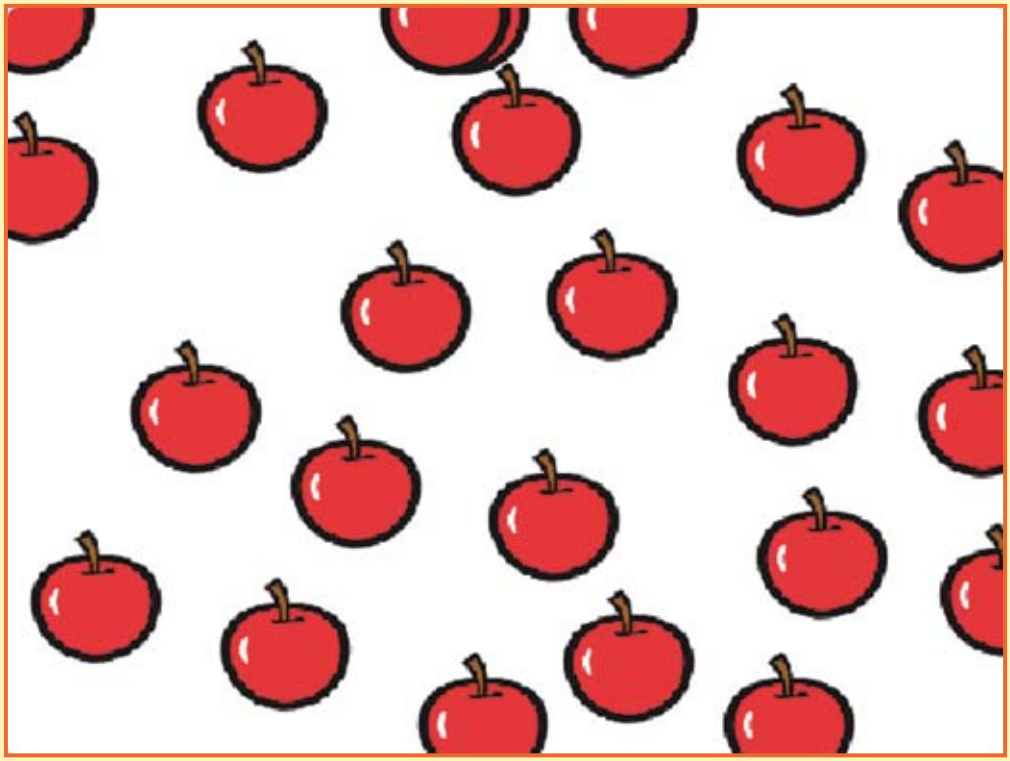
В этой программе используются сразу три новых блока: перейти на указатель мыши, печать и ждать. Первый скрипт делает так, чтобы Яблоко всегда перемещалось вслед за указателем мыши, а вто-

рой всегда отпечатывает на сцене изображение яблока, проигрывает звук и ждёт 1 секунду, чтобы яблоки не сыпались слишком быстро.

Совет

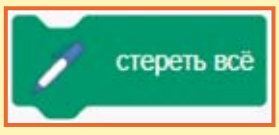
Блок печать появится, если добавить расширение Перо.

Протестируйте работу программы.



Совет

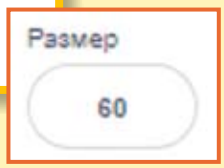
Для того чтобы убрать яблоки со сцены, щёлкните по зелёному блоку стереть всё.



Разбрасывать яблоки на белом фоне не очень интересно. Добавьте на сцену какой-нибудь фон. Например вот этот.

Garden-rock

Уменьшите размер яблока, введя 60 в окне **Размер**.



Теперь можете разбросать яблоки на травке.

Если вы будете аккуратны, то сможете нарисовать красивую рамку или узор.

Проект готов. Сохраните его.

6.7. Задания

1. Измените фон сцены в проекте про яблочки.

2. Сделайте так, чтобы отпечатываемое яблоко изменяло цвет.

3. Сделайте так, чтобы отпечатываемое яблоко изменяло прозрачность.

4. Поэкспериментируйте с временем задержки. Вводите числа с точкой: 0.1, 0.2, 0.5, 0.01, 0.02, 0.05 и проанализируйте результат. Введите отрицательное число, но будьте осторожны, не улетите в прошлое!

5. Измените звук.

6. Сделайте так, чтобы отпечаталось только 15 яблок.

7. Измените костюм Яблока на любой другой из библиотеки спрайтов.

8. Добавьте Яблоку несколько костюмов, которые будут отпечатываться по очереди.

9. Нарисуйте толпу людей.

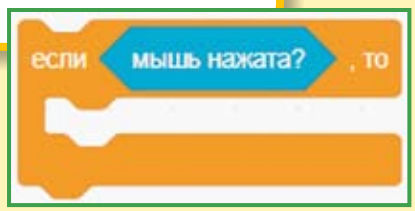
ГЛАВА 7. УСЛОВНЫЙ БЛОК

7.1. Знакомство с условным блоком

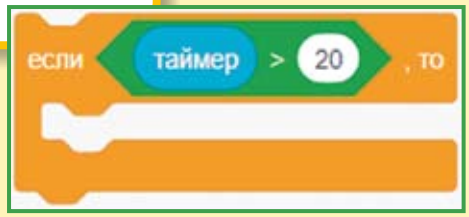
В играх и сложных проектах постоянно происходит множество событий. Например, герой перешёл на следующий уровень, герой нашёл аптечку, враг коснулся героя. При наступлении любого события программа должна выполнить определённые действия, и в этом ей помогает блок проверки условия если.

У этого блока внутри есть специальное шестиугольное углубление, в которое можно вставить шестиугольные зелёные операторы или голубые сенсорные блоки.

Тогда блок если будет выглядеть вот так.



Или так.

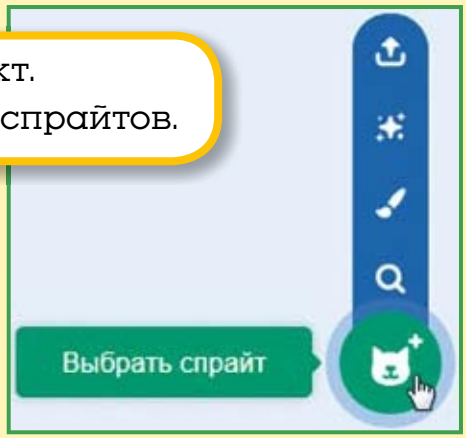


Давайте же скорее применим этот блок!

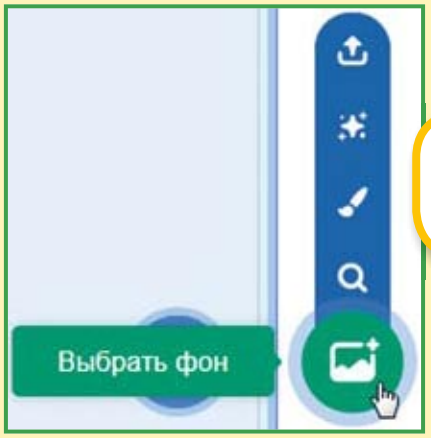
7.2. Игра «Погоня»

Жил да был Кот. Как-то раз, гуляя по городу, он повстречал Синего Пса. Пёс был очень зол и погнался за Котом, желая задать ему хорошую трёпку. Спасётся ли Кот, будет зависеть только от вас!

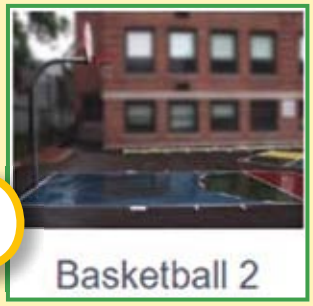
Создайте новый проект.
Откройте библиотеку спрайтов.



Выберите
Синего Пса.



Теперь добавьте
фон на сцену.



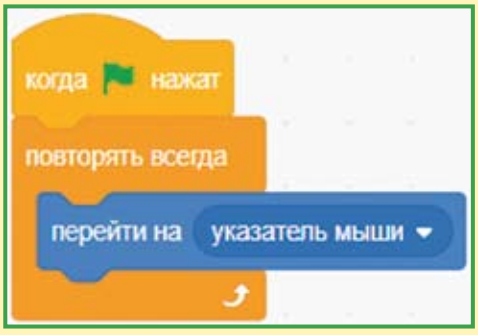
Выберите город.

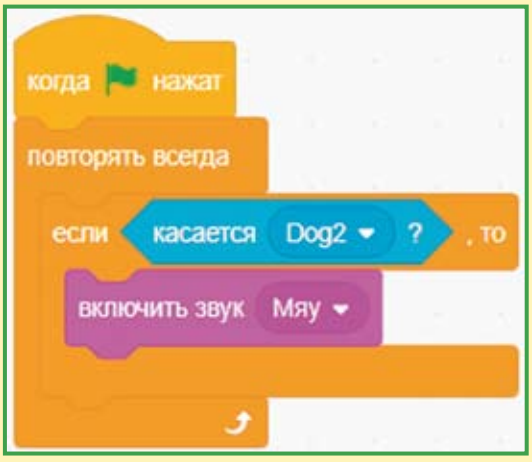
Теперь игра
выглядит вот так.

В проекте всего два спрайта с именами Спрайт 1 и Dog2.

Начинаем программировать!

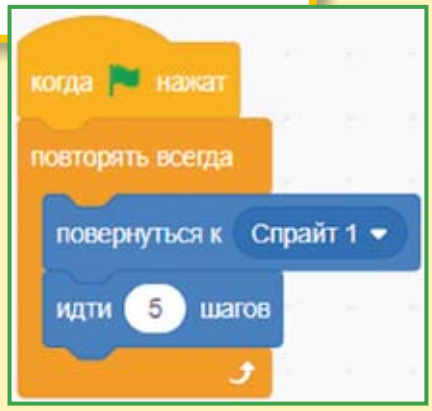
Сначала создадим программу Кота. Она очень простая — состоит из двух скриптов.

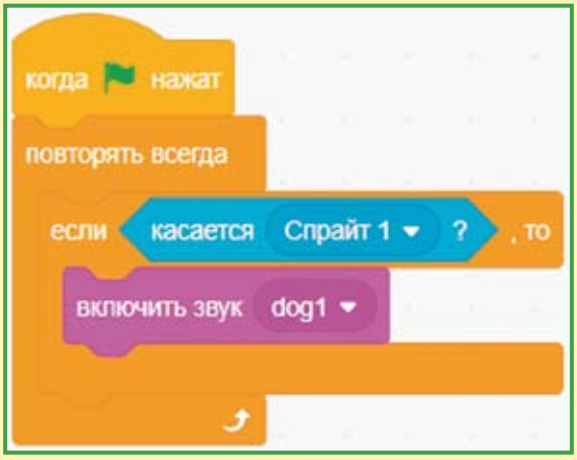




Первый скрипт такой же, как и у Яблока из прошлого проекта. При нажатии на зелёный флажок Кот всегда будет следовать за указателем мыши. Второй скрипт использует новый блок если. Всегда, если Кот коснётся Dog2 (Пса), то жалобно мяукнет, потому что касание вряд ли было приятным.

Программа Синего Пса вот такая.





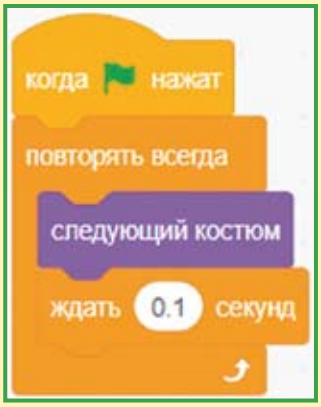
При нажатии на зелёный флажок Синий Пёс всегда будет поворачиваться к Коту и идти по 5 шагов. Если Кот не убежит, то очень скоро попадётся в лапы Пса. Второй скрипт очень похож на скрипт Кота. Всегда, если Синий Пёс коснётся Спрайта 1 (Кота), то он залает.

Протестируйте игру. Убежать от Пса почти невозможно.

7.3. Доработка игры

Кажется, Кот и Синий Пёс не бегают по сцене, а плавают. Давайте добавим анимацию!

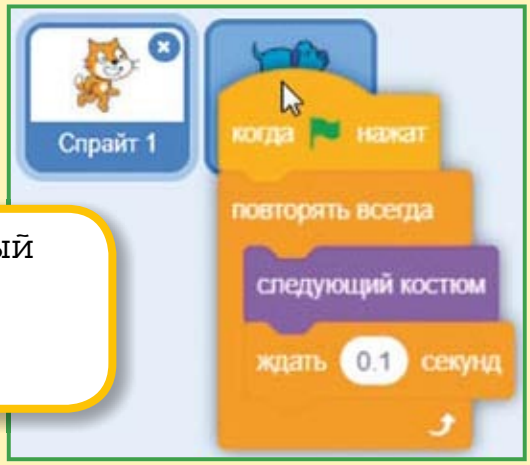
Добавьте Коту вот такой скрипт.



Теперь Кот всегда будет изменять свой костюм через каждую десятую долю секунды.

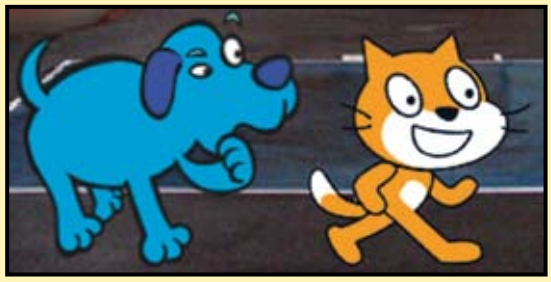
Для Пса сделайте такой же скрипт. Для упрощения задачи можно скрипт Кота скопировать Синему Псу.

Захватите мышью готовый скрипт, тащите прямо на спрайт Dog2 и там отпустите.



Отпускать скрипт надо тогда, когда курсор будет точно находиться над Псом. Готово! Это гораздо проще, чем собирать скрипт из блоков.

Протестируйте работу программы.



Что происходит?! Пёс делает один шаг и на секунду застывает в задумчивой позе! Но он же не философ! Эту ошибку (баг — от английского слова bag, что в переводе означает жук) надо исправить.

Перейдите на вкладку **Костюмы**. У Пса, в отличие от Кота, не два, а три костюма!

Последний костюм для погони не предназначен. Удалите его, нажав на крестик.

Теперь у Синего Пса только два беговых костюма. Программа будет работать правильно.

Сохраните её.

7.4. Задания

1. Усложните игру. Ускорьте бег Синего Пса.

2. Сделайте так, чтобы после касания Кота Пёс, задумавшись, ждал 1 секунду.

3. Добавьте в проект Льва и сделайте так, чтобы при касании Льва Кот мяукал, а Пёс лаял.

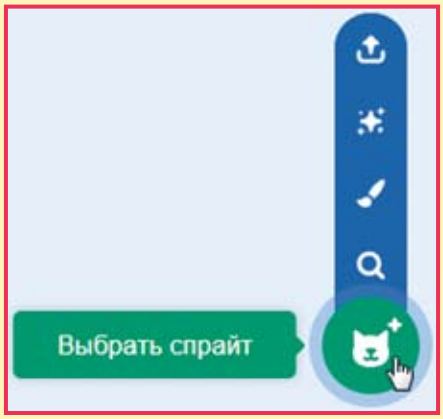
4. Сделайте проект про Кота, который будет управляться клавишами-стрелками вправо и влево и при движении направо будет заводить песнь с помощью блока играть звук, а при движении налево будет говорить сказку с помощью блока говорить.

ГЛАВА 8. МУЛЬТФИЛЬМ «АКУЛА И РЫБКА»

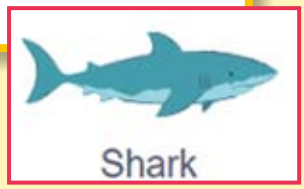
В синем море жили Рыбка и Акула, и как-то раз они повстречались. А что произойдёт дальше, будет зависеть от того, как вы их запрограммируете.

8.1. Создаём персонажей

Запустите Scratch. Добавьте два новых спрайта, нажав кнопку **Выбрать спрайт**.

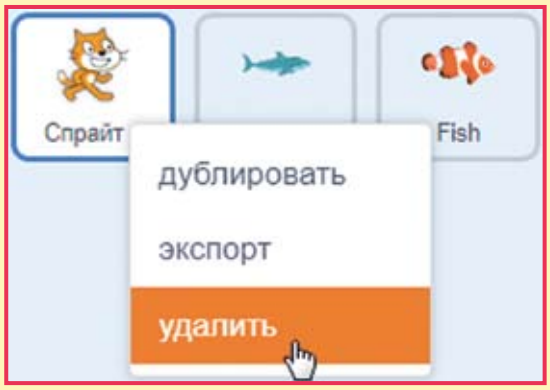


Выберите Акулу (Shark).



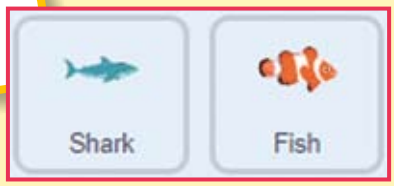
Затем выберите Рыбку (Fish).

Теперь у нас есть три спрайта.

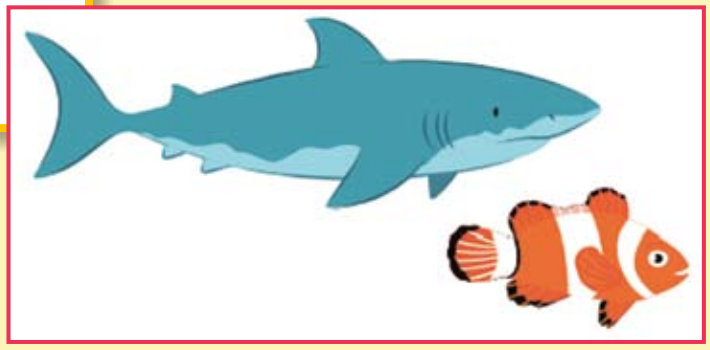


Кота в нашем мультфильме не будет, удалите его. Для этого щёлкните на спрайте Кота правой кнопкой мыши и выберите команду **удалить**.

Остались только рыбы.

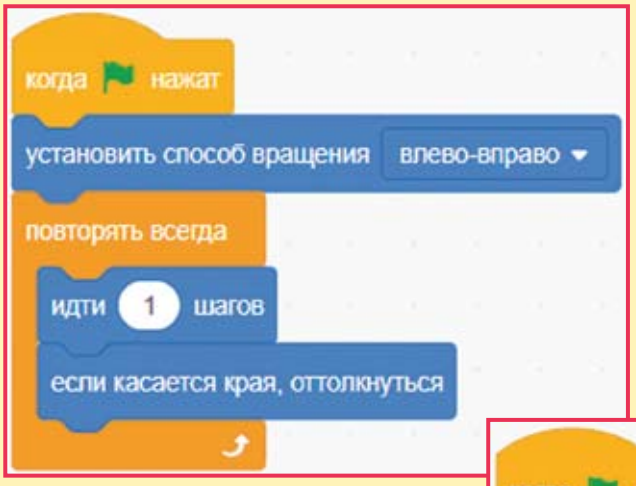


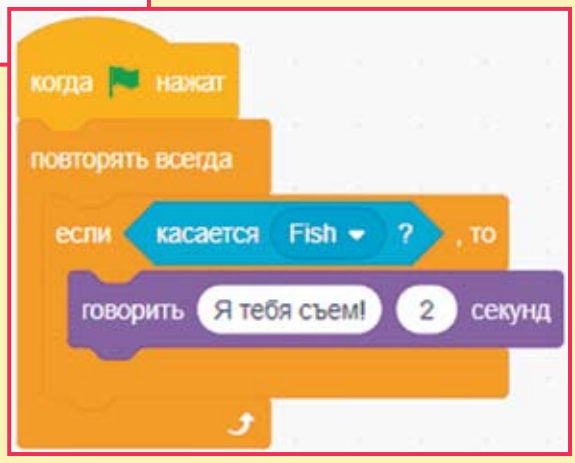
Расположите Акулу повыше, а Рыбку пониже.



8.2. Программируем Акулу

Рыбка будет находиться на месте, а Акула плавать и приговаривать «Я тебя съем!» в момент касания Рыбки. Сначала сделаем программу для Акулы.



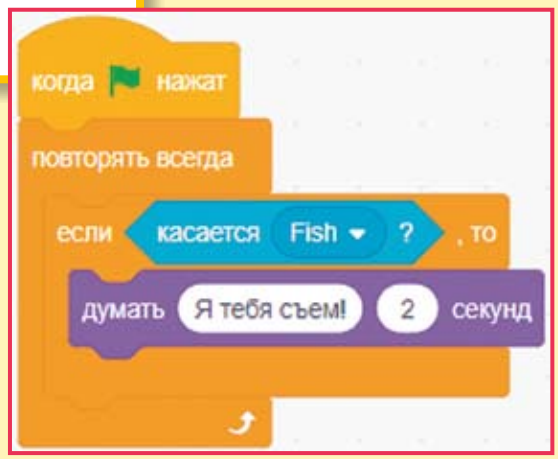


Программа Акулы состоит из двух скриптов. Скрипты запускаются при нажатии на зелёный флажок и работают одновременно. Верхний скрипт отвечает за движение Акулы, а нижний — за касание Рыбки.

Ой! Кажется, рыбы не разговаривают…

Но думать-то они умеют! Давайте заменим блок говорить блоком думать.

Второй скрипт примет такой вид.

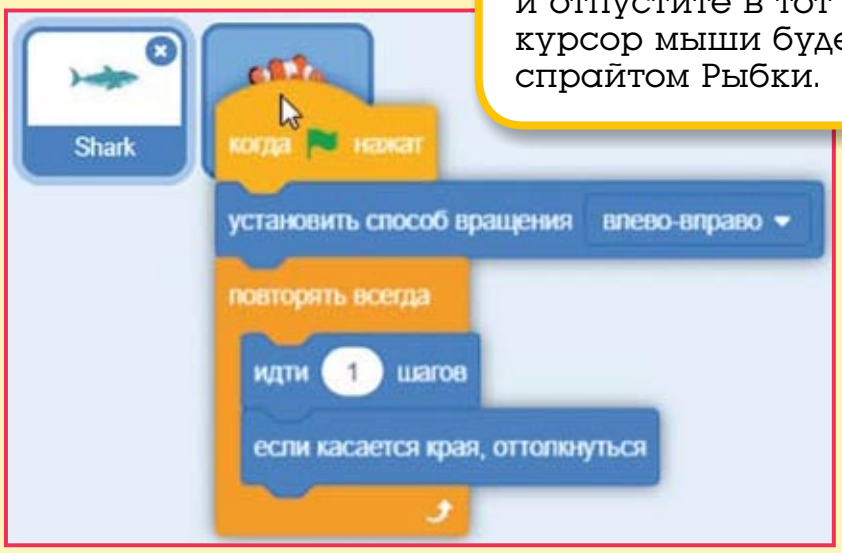


Нажмите на зелёный флажок — Акула поплывёт. Делает она это очень медленно, по одному шагу за раз. Если Акула плавает и ничего не думает, значит, она не касается Рыбки. Немного подвиньте персонажей, чтобы они соприкасались.

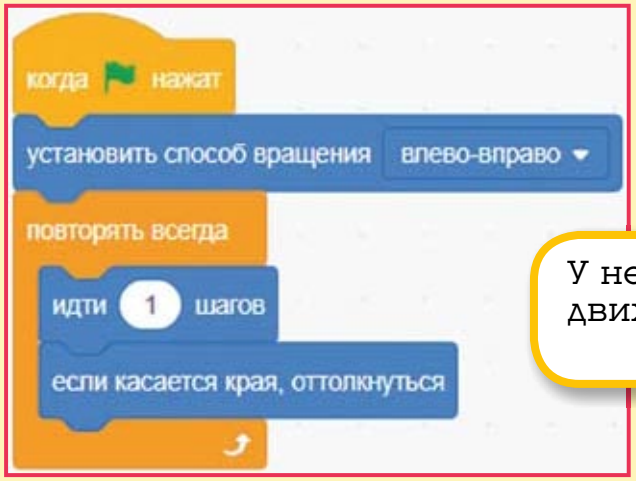
8.3. Программируем Рыбку

Надо дать Рыбке шанс на спасение. Пусть она тоже плавает. Скопируйте ей скрипт с движением Акулы.

Нажмите на верхний блок скрипта движения, тащите его на Рыбку и отпустите в тот момент, когда курсор мыши будет точно над спрайтом Рыбки.



Теперь выберите спрайт Рыбки.



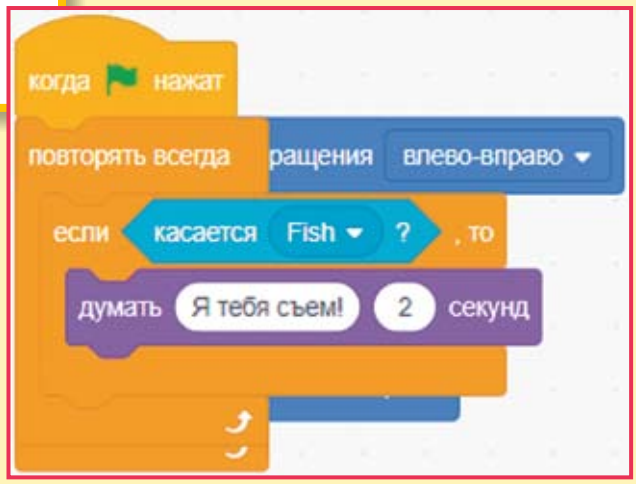
У неё появился такой же скрипт движения, как и у Акулы.

Запускаем программу. Красота! Все плавают! Но Рыбка никак не реагирует, когда её кусает Акула. Давайте скопируем ей и разговорный скрипт от Акулы.

Выберите спрайт Акулы, тащите второй скрипт на Рыбку и отпустите в тот момент, когда курсор мыши будет точно над спрайтом Рыбки.

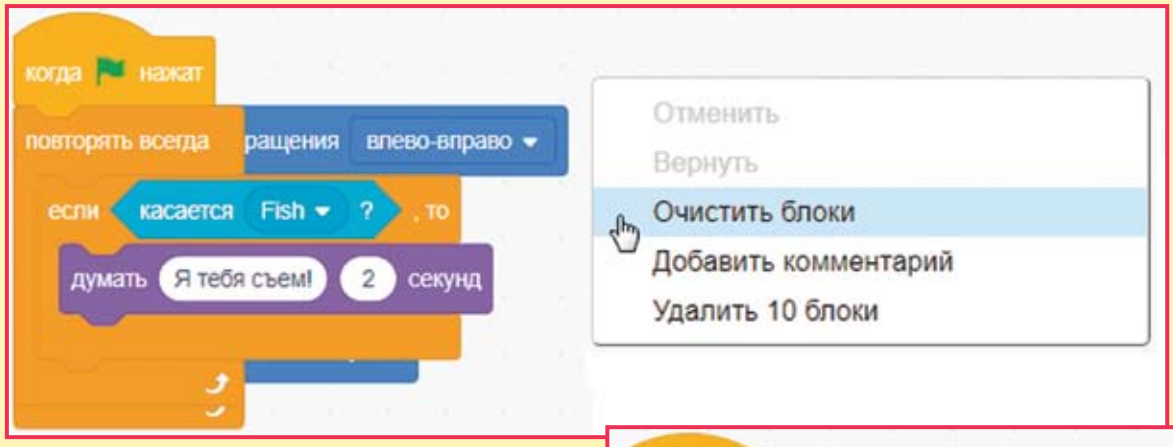
Снова выберите спрайт Рыбки.

Кажется, скрипты смешались в кучу, второй скрипт лежит поверх первого!

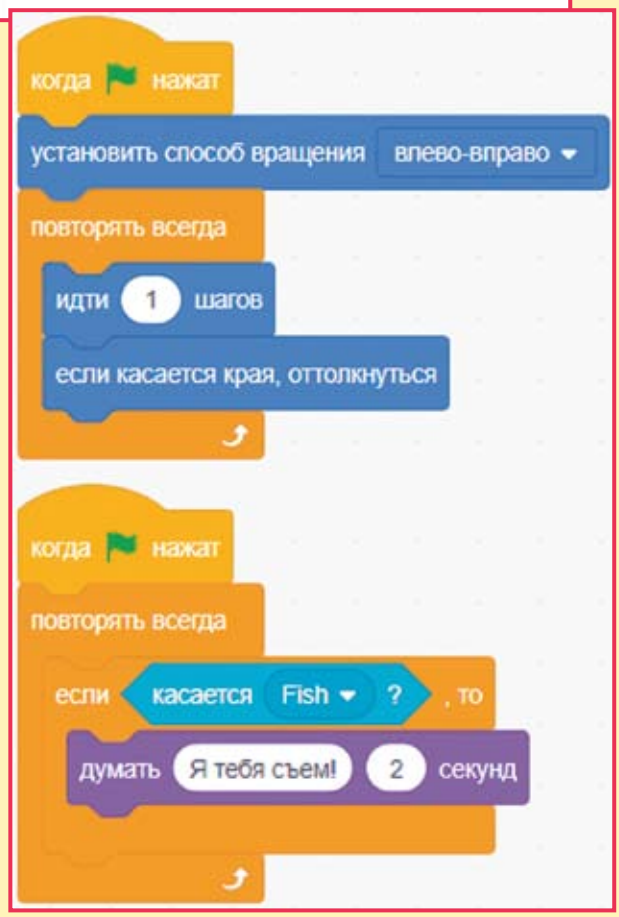


Совет

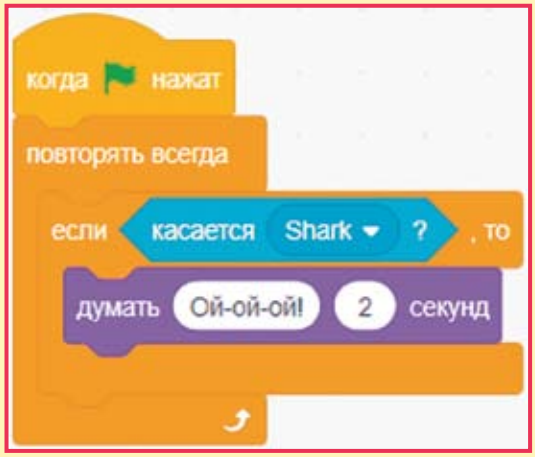
Для того чтобы привести расположение скриптов в порядок, есть хитрый способ. Щёлкните правой кнопкой мыши в пустом пространстве и выберите команду Очистить блоки.

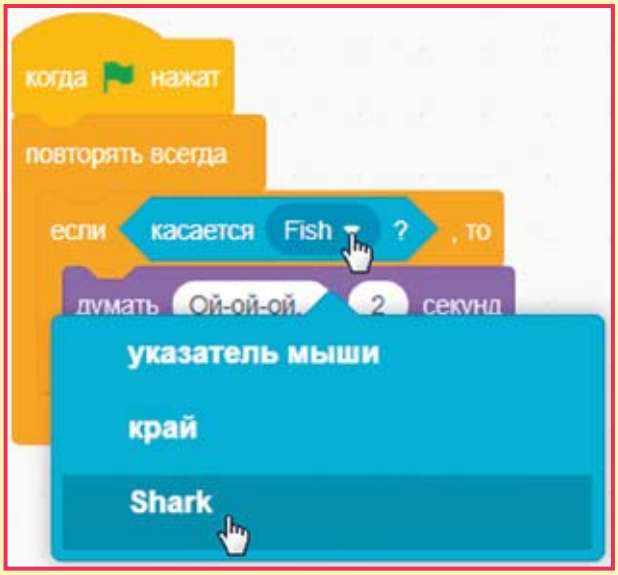


Скрипты расположились ровно. Размещение скриптов не влияет на работу программы.



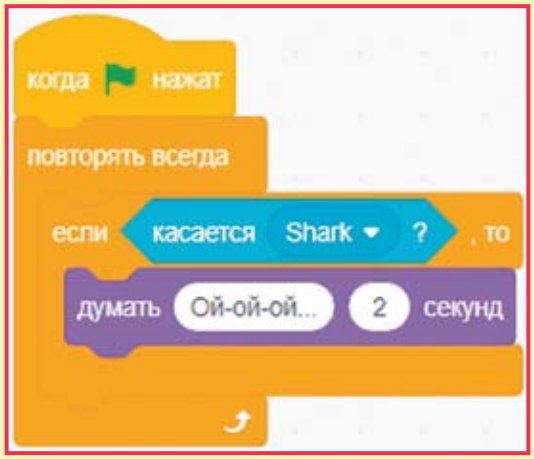
Теперь надо изменить мысль Рыбки. Напишите «Ой-ой-ой!».





А теперь переключите название спрайта в сенсорном блоке касается на акулу (Shark).

Скрипт должен стать вот таким, ведь касаться самой себя рыбка не может!



Мультик готов. Не забудьте сохранить его.

8.4. Задания

1. Добавьте морской фон.

2. Сделайте так, чтобы Акула плавала в 2 раза быстрее рыбки.

3. Добавьте в скрипты движения блоки поворота, чтобы движение стало беспорядочным.

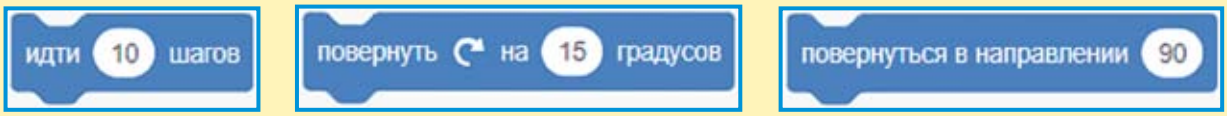
4. Добавьте в мультик ещё одну рыбку, которая будет плавать по диагонали и вежливо здороваться со всеми при касании.

5. Добавьте аквалангиста, который будет молча плавать внизу и останавливаться через каждые 100 шагов на 3 секунды для того, чтобы полюбоваться красотой подводного мира.

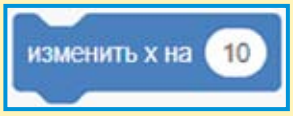
ГЛАВА 9. ЧТО ТАКОЕ КООРДИНАТЫ X И Y

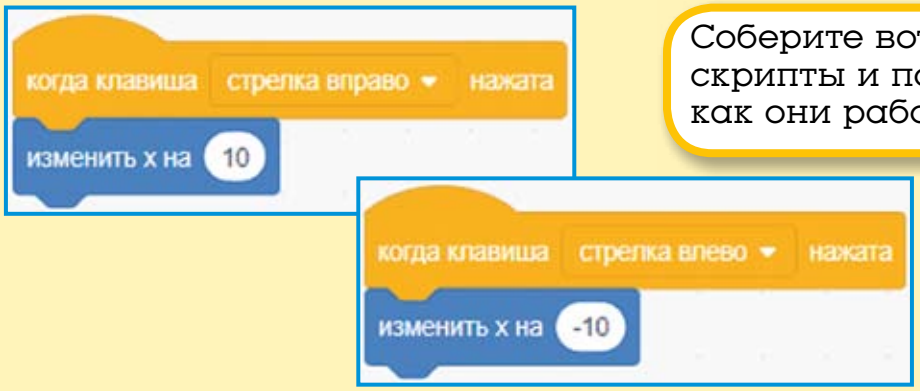
9.1. Перемещение по горизонтали

Вы уже знаете, что для перемещения спрайтов по экрану существуют синие блоки идти, повернуть и повернуться в направлении.



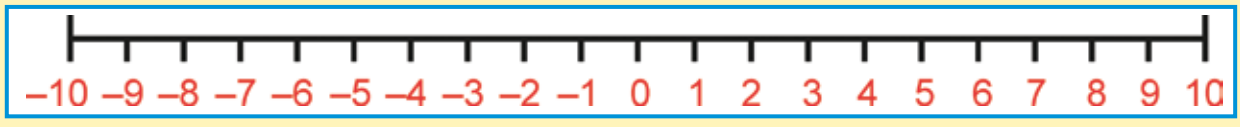
Эти блоки позволяют переместить спрайт в любую точку экрана. Однако для перемещения спрайта в горизонтальном направлении (справа налево или слева направо) лучше использовать специальный блок изменить x.



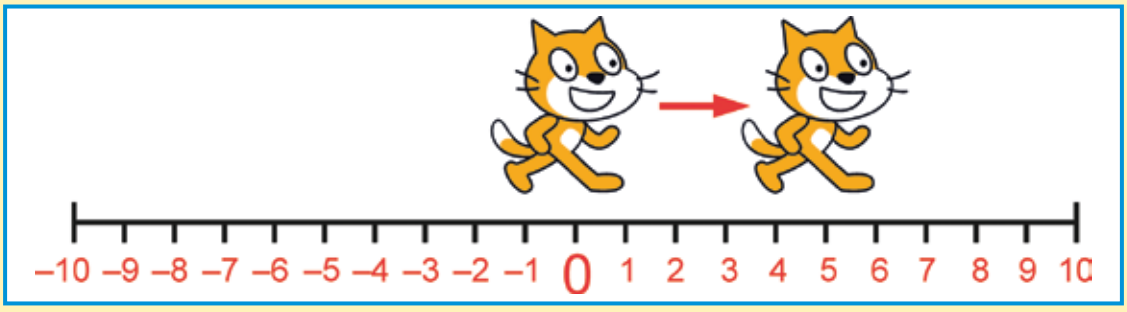


Соберите вот такие скрипты и посмотрите, как они работают.

Спрайт управляется стрелками и перемещается в горизонтальном направлении, но почему это происходит? Для того чтобы понять, как работают эти скрипты, нужно вспомнить про *координатную ось x* (произносится как «ось икс»).

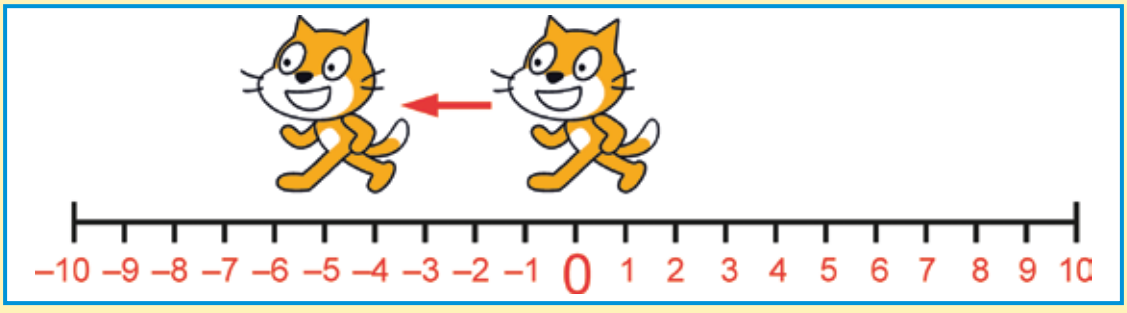


Ось x — это горизонтальная линия, состоящая из множества точек, каждая из которых имеет свой номер. Центр оси x находится в середине сцены, в этой точке координата X равна нулю. Значения X справа от центральной точки — положительные, а слева — отрицательные. Если спрайт расположен в центре экрана и его координата X равна нулю, а затем он получит команду изменить x на 5, то спрайт переместится на 5 шагов вправо.

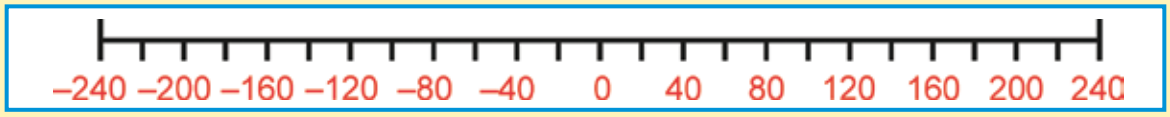


Таким образом, получается, что следующие скрипты выполнят одно и то же действие.

В случае же если спрайт расположен в центре экрана, а затем он получит команду изменить x на -5, то спрайт переместится на 5 шагов влево.



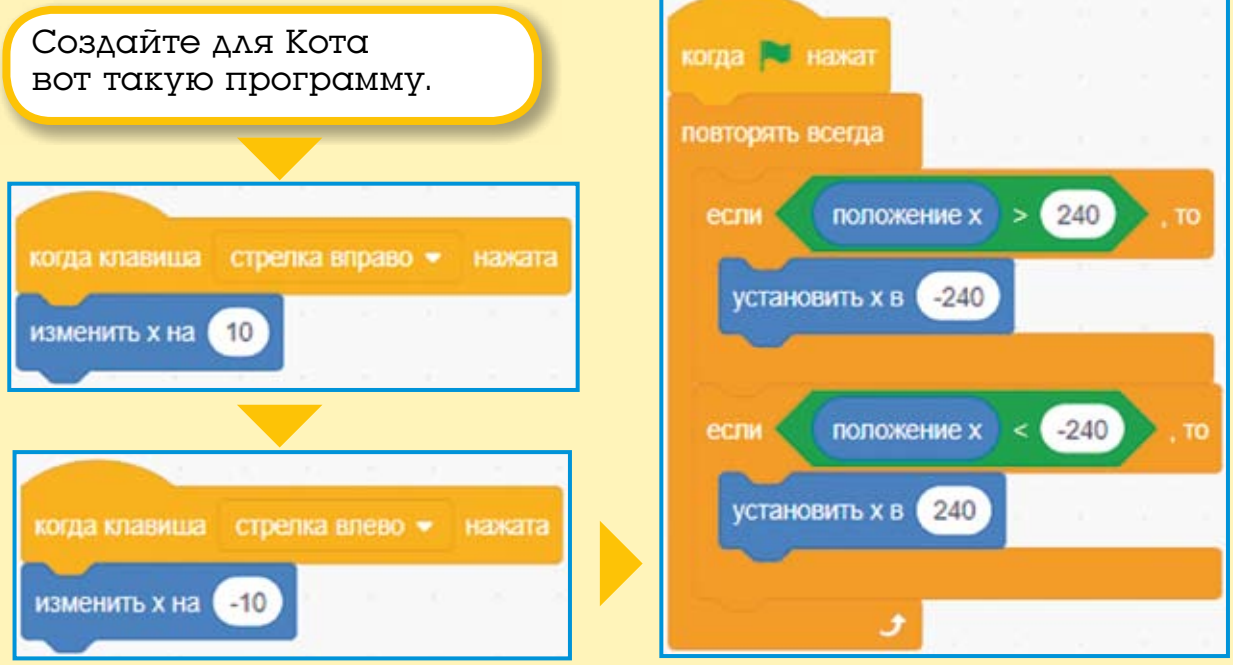
Координата X в Scratch может изменяться от -240 до 240.



Если дать спрайту задание перейти в точку с координатой X больше 240, то спрайт скроется за краем экрана.

Движение с помощью координаты X позволяет более гибко управлять перемещением спрайтов. Например, можно сделать так, чтобы Кот, который дошёл до правой границы сцены, появлялся слева и продолжал движение направо. И наоборот, Кот, который дошёл до левой границы, появлялся справа и продолжал движение налево.

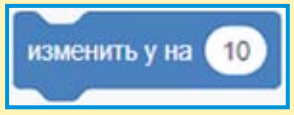
Создайте для Кота вот такую программу.



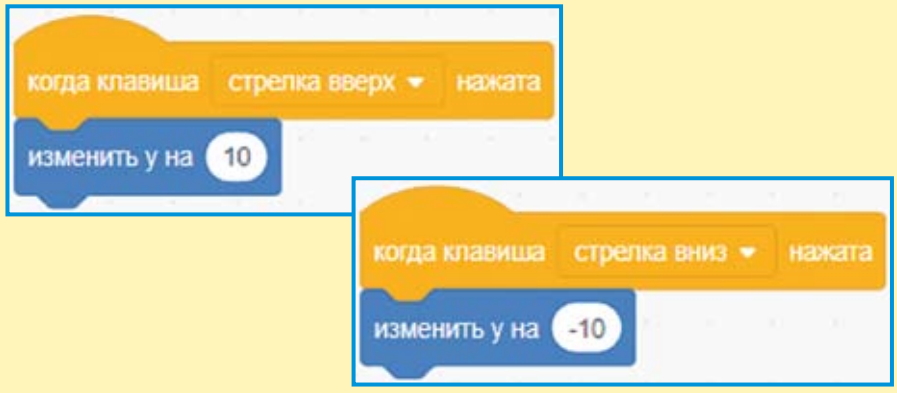
Обратите внимание, здесь использован новый блок установить x, он позволяет «телепортировать» спрайт в нужную точку. В нашем случае спрайт «телепортируется» на противоположную сторону сцены. Овальный блок положение x — это *защищённая переменная*, которая всегда хранит значение координаты *X* спрайта. Подробнее о переменных вы узнаете в следующих главах.

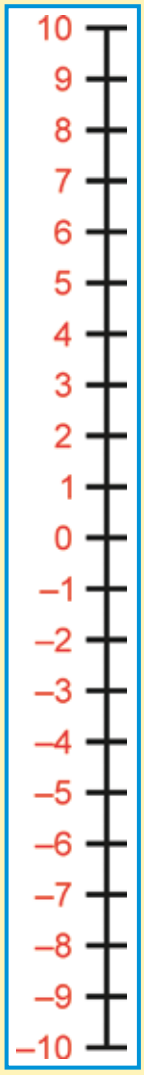
9.2. Перемещение по вертикали

Для перемещения спрайтов по вертикали существует блок изменить y.



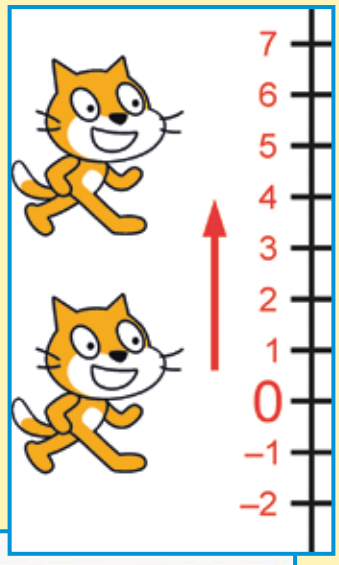
Сделайте вот такие скрипты и посмотрите, как они работают.



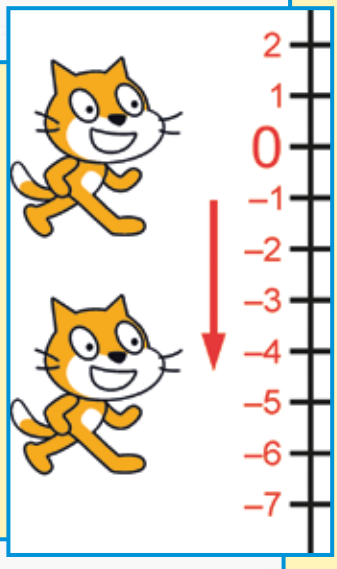


Спрайт управляется стрелками и перемещается в вертикальном направлении. Для того чтобы понять, как работают эти скрипты, нужно вспомнить про *координатную ось y* (произносится как «ось игрек»). Ось *y* очень похожа на ось *x*, она тоже состоит из множества пронумерованных точек. Её центр находится в середине сцены. Есть только одно отличие — ось *y* расположена вертикально.

Значения *Y* сверху от центральной точки — положительные, а снизу — отрицательные. Если спрайт расположен в центре экрана и его координата *Y* равна нулю, а затем он получит команду изменить у на 5, то спрайт переместится на 5 шагов вверх.

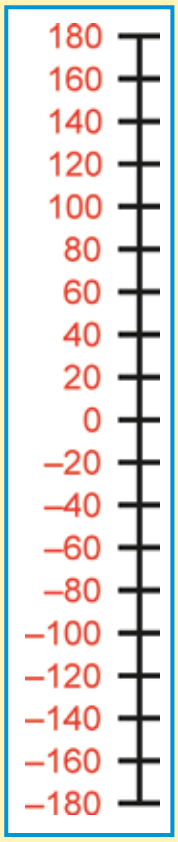


Таким образом, получается, что следующие скрипты выполнят одно и то же действие.



В случае же если спрайт расположен в центре экрана, а затем он получит команду изменить у на -5, то спрайт переместится на 5 шагов вниз.

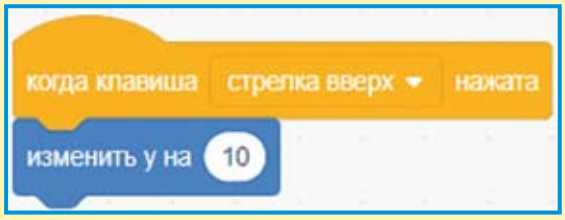
То есть эти скрипты эквивалентны.

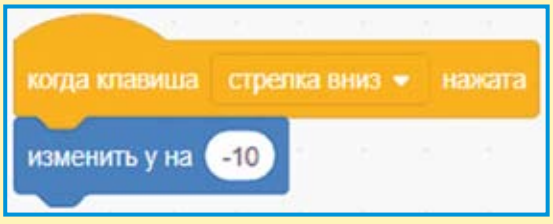


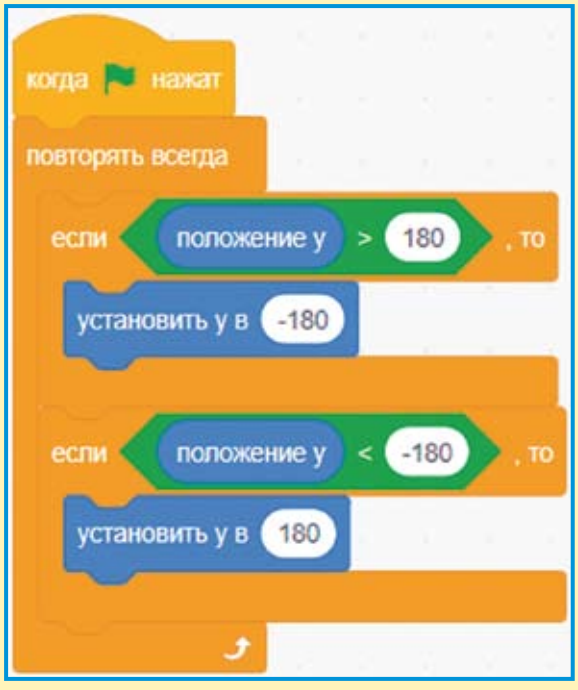
Координата *Y* в Scratch может изменяться от –180 до 180.

Если дать спрайту задание перейти в точку с координатой *Y* больше 180 или меньше –180, то спрайт, как вы, наверное, догадались, скроется за краем сцены.

Добавьте в программу Кота вот такие скрипты.







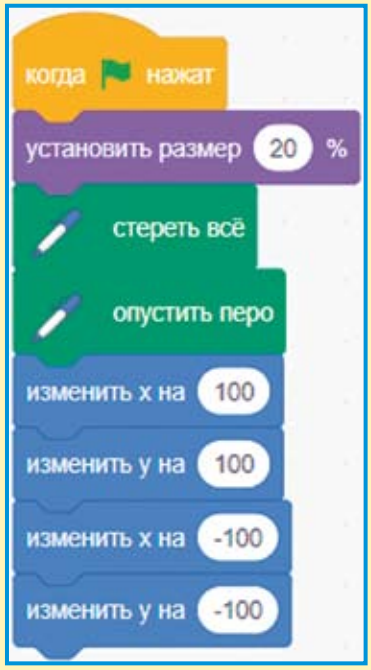
Теперь, если Кот поднимется до верхней границы сцены, то он появится снизу и продолжит движение вверх. А если Кот опустится до нижней границы, то он появится сверху и продолжит движение вниз.

Обратите внимание, здесь использован новый блок установить y. Он работает так же, как и блок установить x, и позволяет «телепортировать» спрайт в нужную точку. В нашем случае спрайт «телепортируется» на противоположную сторону сцены. Овальный блок положение y — это *защищённая переменная*, которая всегда хранит значение координаты *Y* спрайта. Подробнее о защищённых переменных вы узнаете в следующих главах.

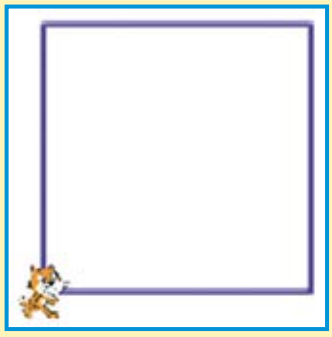
9.3. Рисование по координатам

Теперь, когда вы знакомы с координатами *X* и *Y*, можно попробовать нарисовать что-нибудь на сцене с применением новых блоков.

Создайте новый проект и соберите для Кота вот такой скрипт.

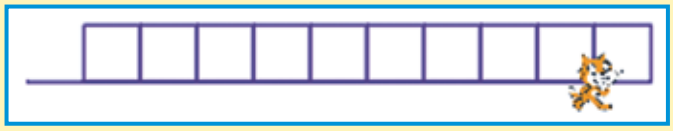


При запуске программы Котик изменит размер и станет в 5 раз меньше обычного (ведь 20% в 5 раз меньше, чем 100%). Сцена очистится, перо опустится, и Кот четыре раза изменит свои координаты так, что получится квадрат.

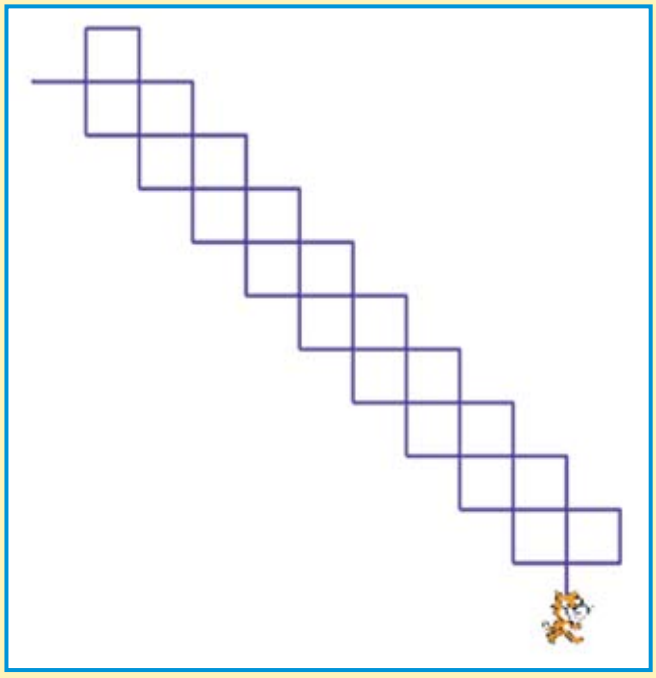


С применением блока повторить можно рисовать очень красивые узоры. Создайте новый проект и сделайте для Кота вот такой скрипт.

Перетащите Кота в центр сцены и запустите скрипт, он нарисует ряд квадратиков.



Измените значение в самом нижнем блоке. Вместо —20 введите —40. Теперь Котик рисует вот такой узор.



Изменяйте значения в блоках движения и нарисуйте свои узоры. Не забывайте вручную перемещать Кота в центр экрана.

9.4. Задания

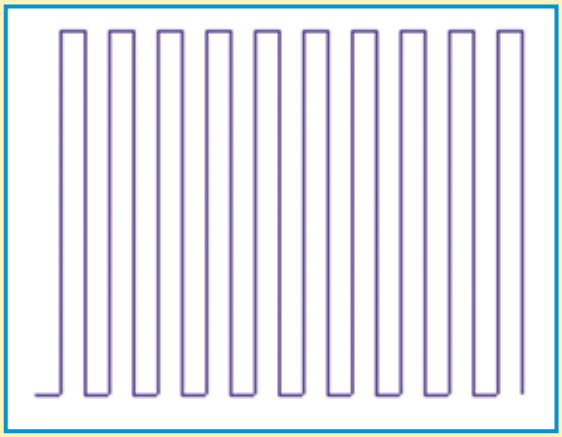
1. Доработайте программу, в которой вы научили Котика рисовать: добавьте скрипт, который по нажатию клавиши <Пробел> очищает сцену, перемещает Кота в центр и восстанавливает его размер.

2. Сделайте так, чтобы при запуске программы Кот автоматически попадал в центр экрана, и вам не приходилось бы перетаскивать его вручную.

3. Измените цвет и толщину пера.

4. На листе тетради «в клеточку» нарисуйте змейку, а затем научите Кота её рисовать.

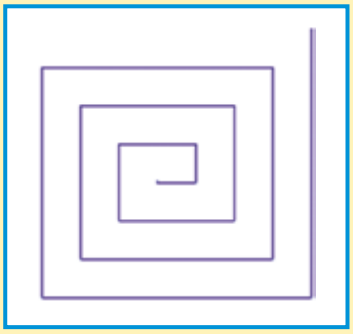
5. На листе тетради «в клеточку» нарисуйте забор, а затем научите Кота его рисовать.



6. На листе тетради «в клеточку» нарисуйте спираль, а затем научите Кота её рисовать.

Подсказка

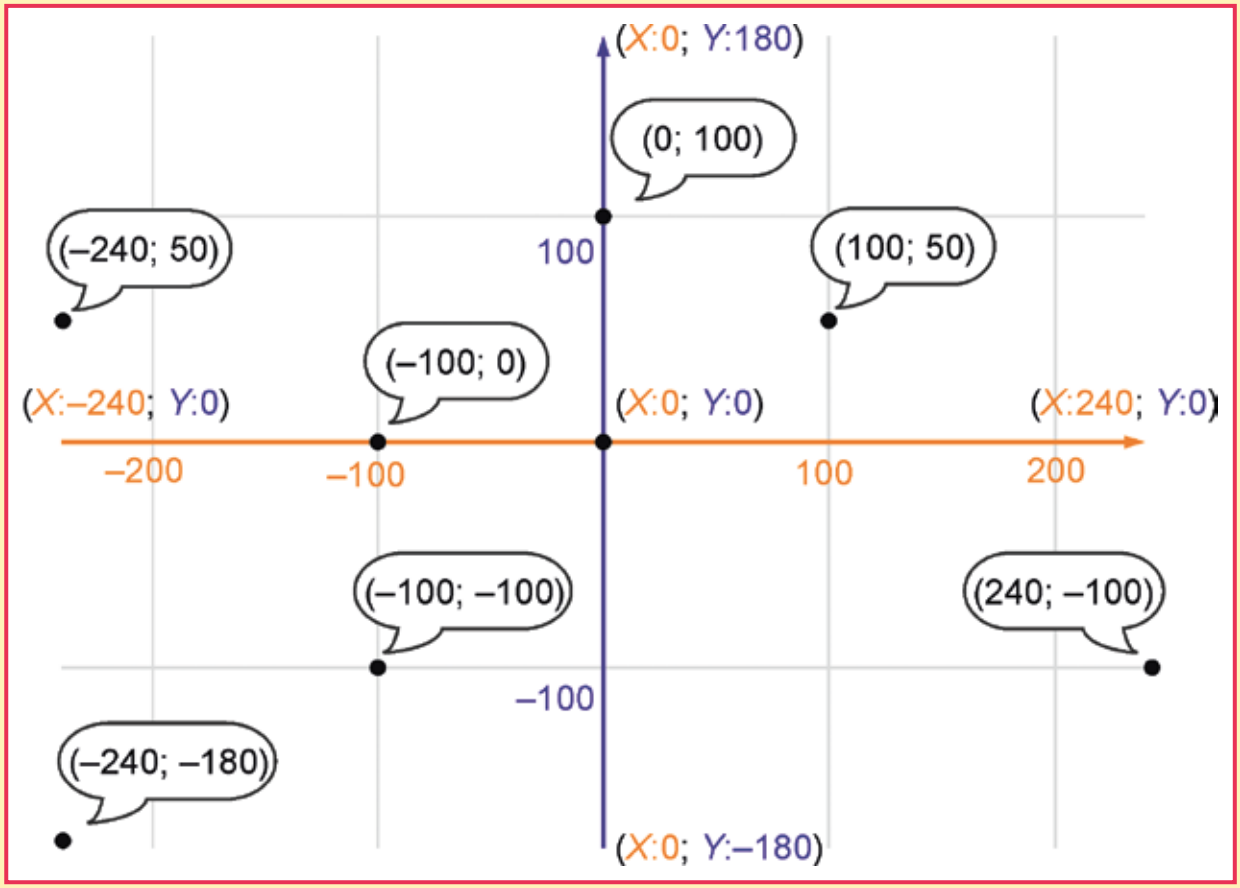
Блок повтора здесь не нужен.



ГЛАВА 10. МУЛЬТФИЛЬМ «ПИКО И ПРИВИДЕНИЕ»

10.1. Координатная плоскость

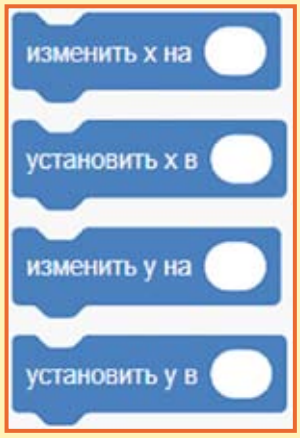
Как вы знаете, монитор компьютера состоит из множества отдельных точек — *пикселов*. Сцена в Scratch также состоит из отдельных точек. Размер сцены 480 точек слева направо и 360 точек снизу вверх. Для того чтобы отличать точки одну от другой, им даны две координаты: X и Y. Каждая точка сцены имеет свои координаты. Координаты записывают в скобках: сначала X, потом Y — $(X; Y)$.



Совет

В области свойств спрайта вы всегда можете увидеть *координаты выбранного спрайта*. Сейчас Котик расположен в точке с координатами (100; 50).

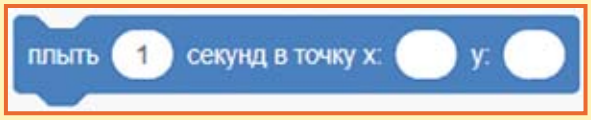
10.2. Новые блоки перемещения по координатной плоскости



Вы уже научились пользоваться некоторыми блоками перемещения по сцене с помощью координат.

Осталось познакомиться ещё с двумя. Блок перейти в x y «телепортирует» спрайт в точку сцены с указанными координатами. Он делает то же самое, что и пара блоков установить x и установить y.

Ещё один блок плыть 1 секунд в точку x y плавно перемещает спрайт в требуемую точку экрана.



Теперь, когда вы знакомы со всеми блоками движения, можно создать мультфильм.

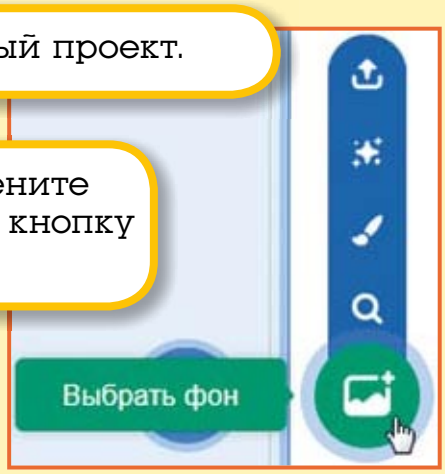
10.3. Делаем мультфильм

Как-то раз Пико гулял в старом замке. Он, как обычно, слушал свой плеер и совсем не заметил, как появилось Привидение! «Мама!» — закричал Пико. Привидение улыбнулось, сказало: «Привет!» — и растаяло.

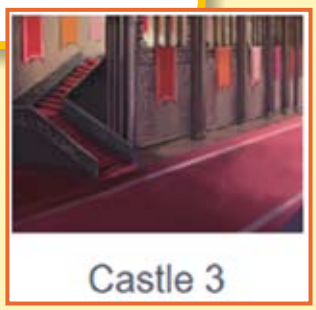
Давайте сделаем маленький мультфильм о приключениях Пико в старом замке.

Создайте новый проект.

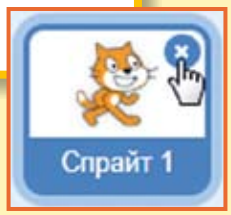
Сначала измените фон. Нажмите кнопку **Выбрать фон**.



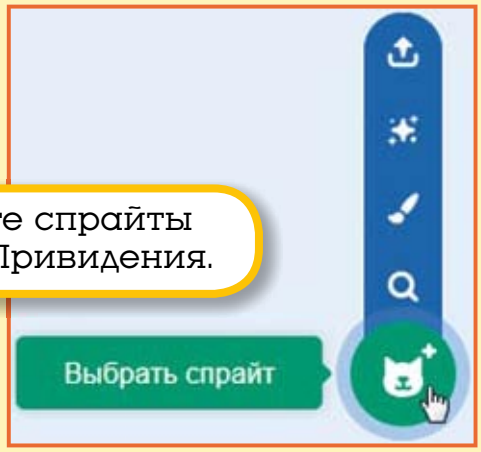
Выберите старый замок.



Кота в нашей истории не будет. Удалите его.



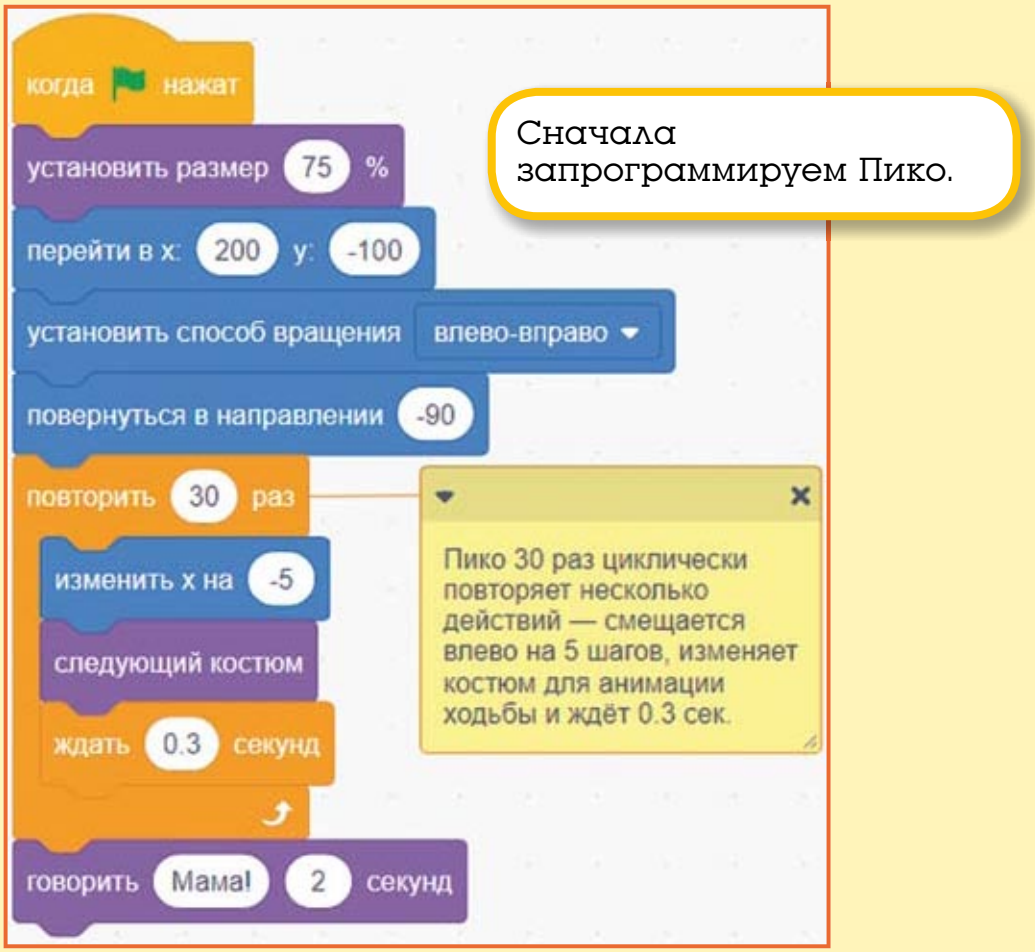
Добавьте спрайты Пико и Привидения.



Теперь у нас есть старый замок, Пико и Привидение.

Начинаем программировать!

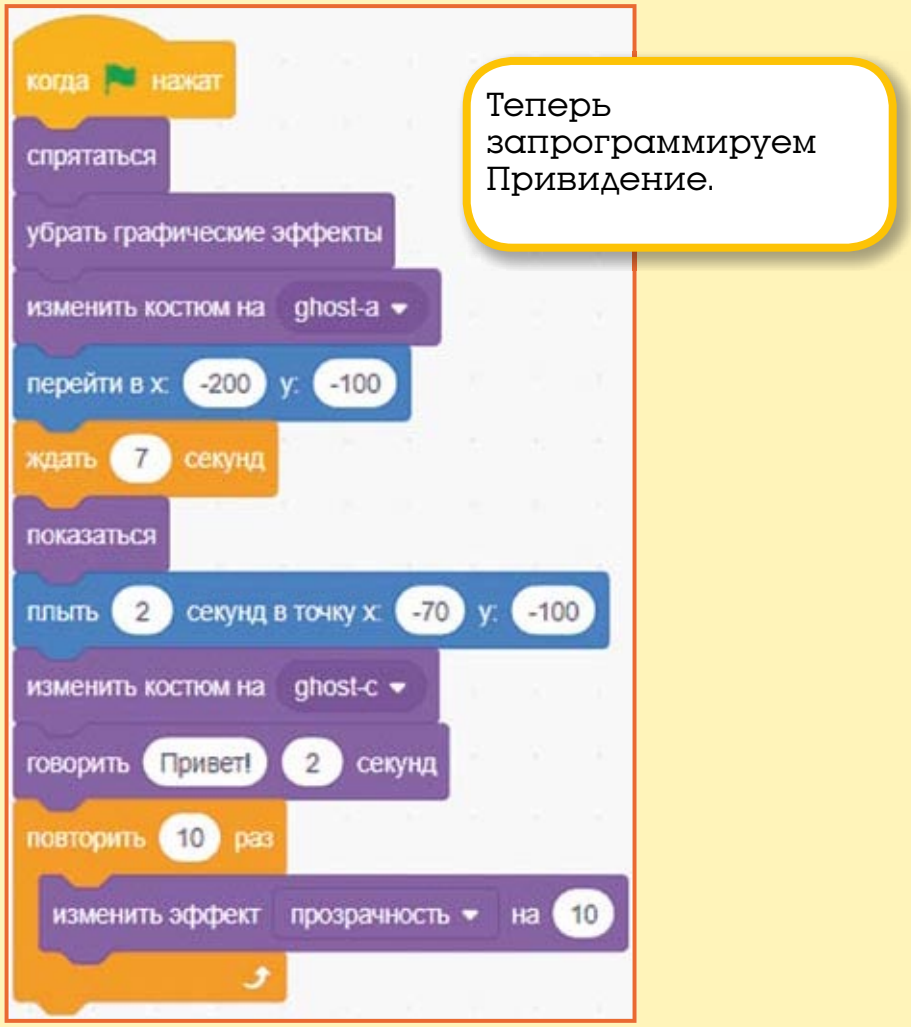
10.4. Программируем Пико и Привидение



Сначала запрограммируем Пико.

Давайте посмотрим, как работает этот скрипт. Сначала Пико немного уменьшит свой размер и перейдёт в правый нижний угол сцены, в точку с координатами (200; –100). Затем мы устанавливаем способ вращения влево-вправо и поворачиваем Пико лицом налево, ведь если такой стиль вращения не задать, то Пико перевернётся вверх ногами! Затем Пико 30 раз циклически повторяет несколько действий — смещается влево на 5 шагов, изменяет костюм для анимации ходьбы и немного ждёт (примерно треть секунды). После того как работа цикла закончится, Пико окажется на 150 шагов левее, чем в начале — он будет почти в центре сцены.

Обратите внимание!
К оранжевому блоку повторить через меню, открываемому правой кнопкой мыши, добавлено примечание. *Примечание* — это пометка, помогающая другим понять, как работает ваш проект. Не забывайте оставлять примечания в своих проектах!



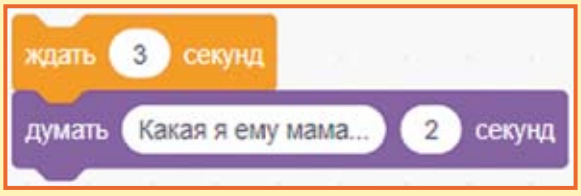
После запуска программы Привидение спрячется, немного уменьшит размер, наденет злой костюм, перейдёт в левый нижний угол экрана и притаится на 7 секунд. Когда терпение Привидения лопнет, оно появится, не спеша поплывёт навстречу

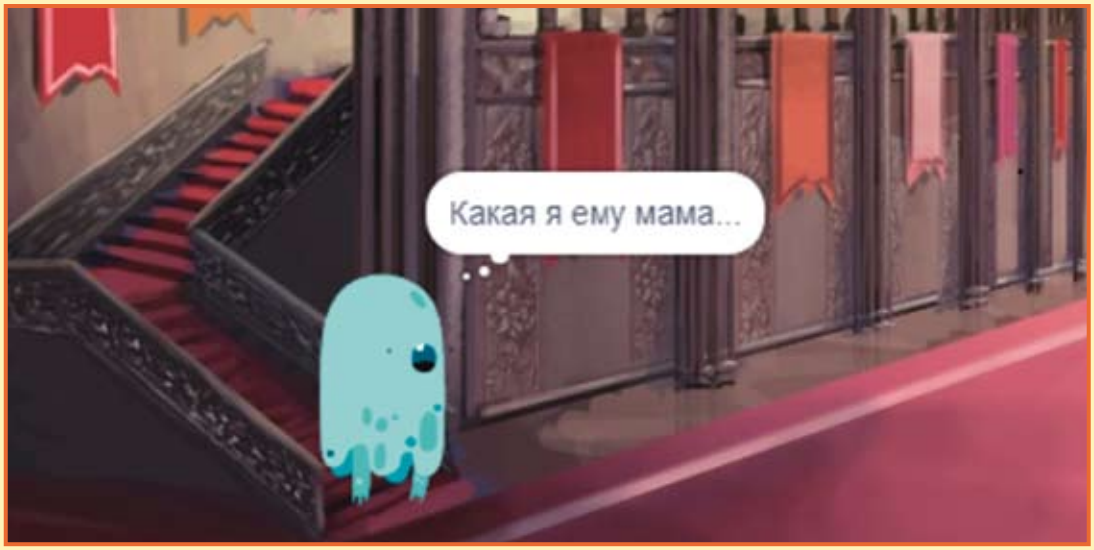
Пико, переоденется в кричащий костюм и скажет: «Привет!», после чего растает, циклически изменив эффект призрака 10 раз на 10.

Проект готов, сохраните его.

10.5. Задания

1. Ускорьте передвижение Пико.
2. Ускорьте передвижение Привидения.
3. А теперь замедлите исчезновение Привидения.
4. Сделайте так, чтобы после исчезновения Привидения Пико продолжил движение до левой границы сцены.
5. Добавьте ещё одно привидение, которое будет летать в верхней части сцены.
6. Добавьте смех Привидению, записав его.
7. Добавьте привидению два блока с фразой в конец скрипта, например, вот таких:





ГЛАВА 11. ИГРА «ЛАБИРИНТ»

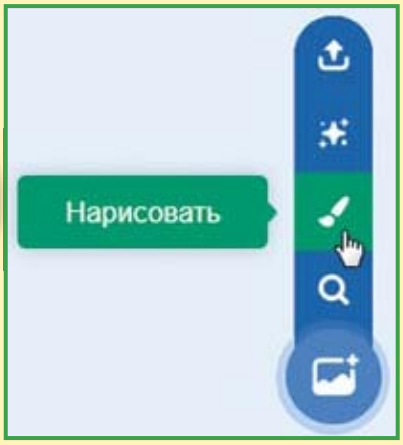
Жил-был вечно голодный Динозавр. Каждое утро он вызывал службу доставки еды, которая спешила накормить его, пока он не наломал дров.

Давайте сделаем игру, в которой Машинке надо проехать через лабиринт и накормить Динозавра. Стен касаться нельзя!

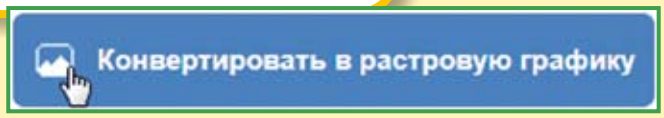
11.1. Рисуем лабиринт

Создайте новый проект.

Начнём с лабиринта.
Нажмите кнопку **Нарисовать**.

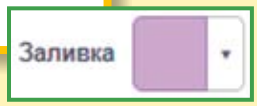


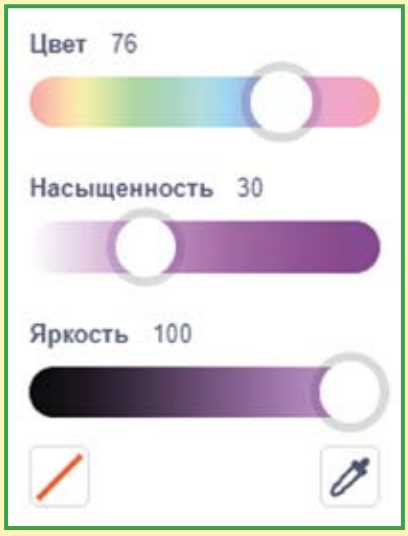
Нажмите кнопку
Конвертировать в растровую графику.



Выберите инструмент **Заливка**.

Выберите цвет 76
с насыщенностью 30
и яркостью 100.

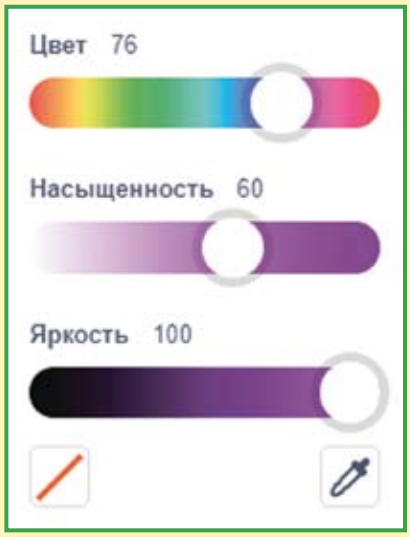




Залейте сцену этим цветом.

Теперь нарисуем стены. Выберите инструмент **Линия**.

Выберите цвет 76 с насыщенностью 60 и яркостью 100.

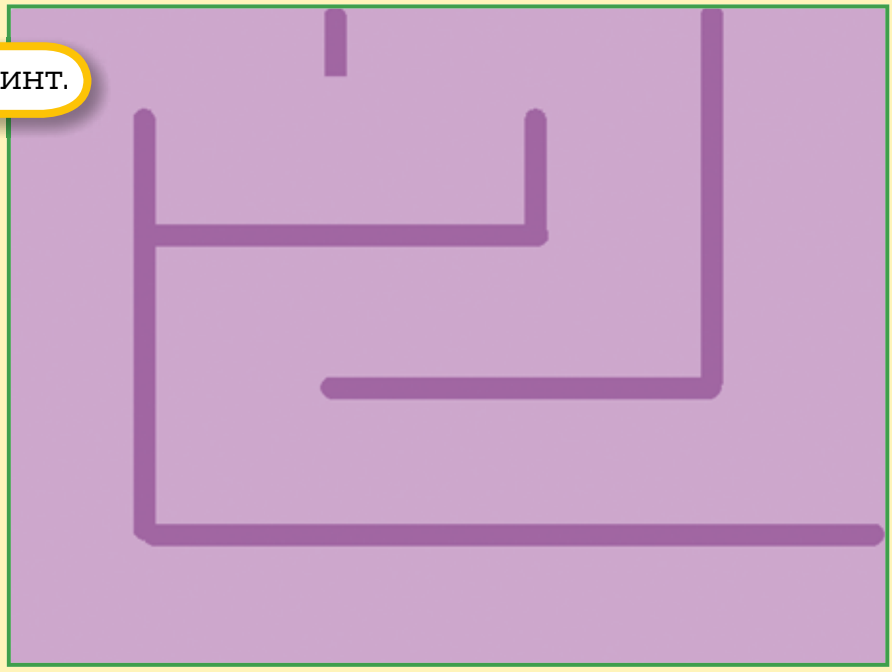


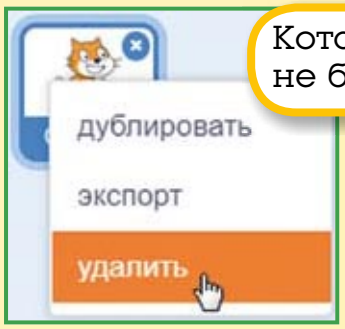
Увеличьте ширину линии до 25.

Совет

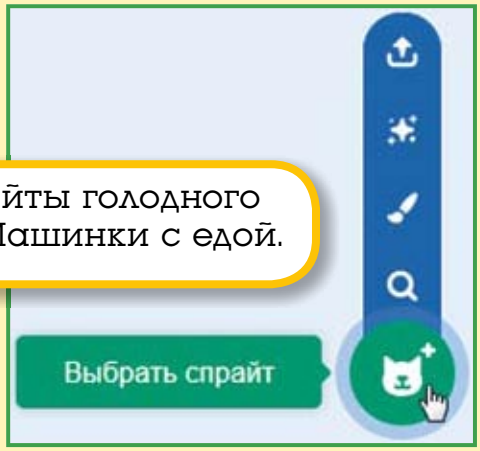
Для того чтобы линии получались ровными, рисуйте при нажатой клавише <Shift>.

Нарисуйте лабиринт.





Кота в нашей истории не будет. Удалите его.



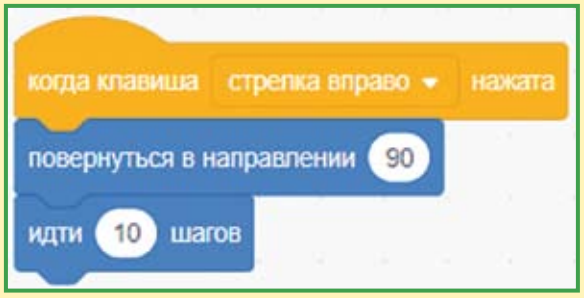
Добавьте спрайты голодного Динозавра и Машинки с едой.

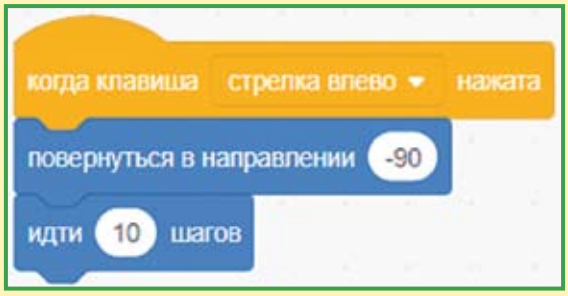
Теперь у нас есть лабиринт, Динозавр и Машинка.

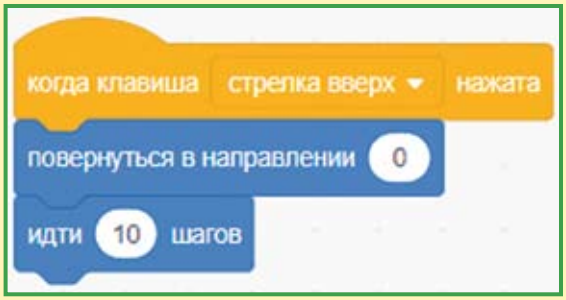
Можно начинать программировать!

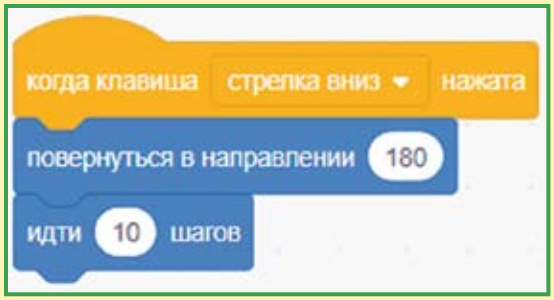
11.2. Программируем спрайты

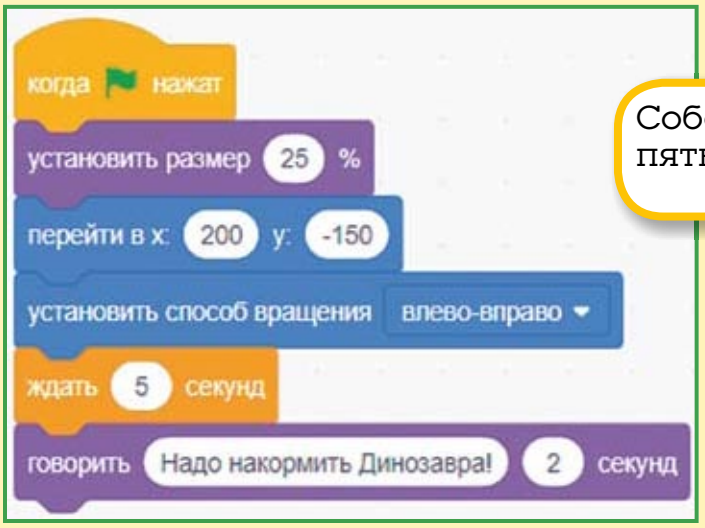
Сначала запрограммируем Машинку. Соберите ей четыре скрипта для управления стрелочками.





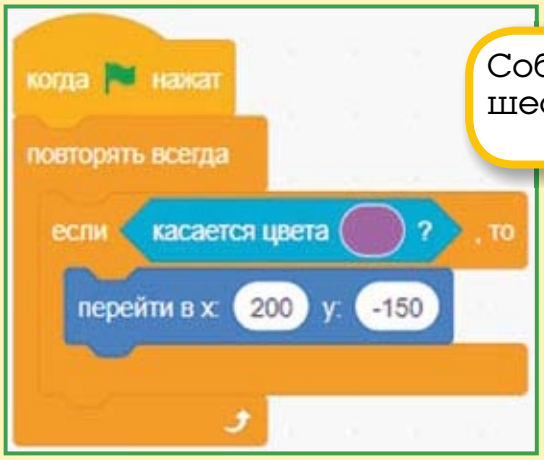






Соберите пятый скрипт.

Рассмотрим, как работает скрипт. Сначала уменьшается размер Машинки, чтобы она смогла протиснуться через лабиринт, она перемещается в правый нижний угол, устанавливает стиль вращения, ждёт 5 секунд и произносит фразу.



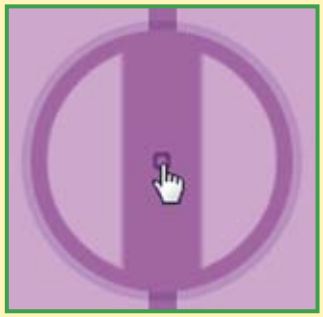
Соберите шестой скрипт.

Машинка в вечном цикле постоянно проверяет, не коснулась ли она стены, и если коснулась, то отправляется на старт.

Для того чтобы выбрать цвет в блоке касается цвета, нужно щелкнуть в овальчике 

и выбрать пипетку  .

Затем выбрать цвет стены на сцене.



Протестируйте работу скриптов Машинки. Что вы заметили? Она едет задом наперёд!

Это всё из-за того, что спрайт Машинки нарисован повёрнутым влево. Давайте перевернём её, чтобы она ездила не на задней скорости.

Выберите спрайт Машинки и перейдите на вкладку **Костюмы**.

Нажмите на инструмент **Выбрать**.

Выделите Машинку рамкой.

Нажмите на инструмент **Отразить по горизонтали**.

Машинка повернётся вправо.

Отлично! Снова протестируйте работу Машинки, она должна ехать носом вперёд.

Теперь запрограммируем Динозавра.

Задача Динозавра очень простая — стой и жди, когда тебя накормят. В начале работы программы Динозавр переходит на место, уменьшается, изменяет костюм и просит привезти ему еду. Затем он ждёт до того момента, пока не коснётся спасительной машинки с едой, и, когда коснётся, станцует весёлый «Танец сытого динозавра», 40 раз переключая костюмы, и программа будет остановлена.

Сохраните проект.

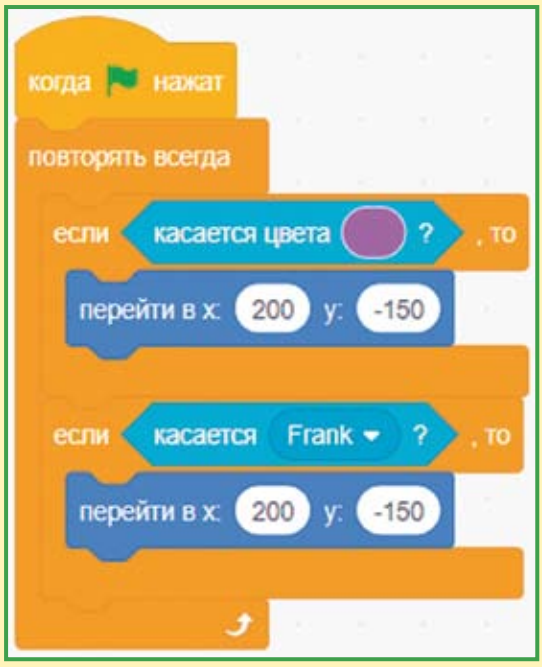
11.3. Усложняем игру

Если вы немного поиграете в эту игру, то поймёте, что она не очень сложная. Для того чтобы сделать проект интереснее, можно добавить в лабиринт персонаж, с которым нельзя встречаться Машинке.

Добавьте из библиотеки спрайт Охранника, например, Френка.

Соберите для него такой скрипт.

Размер Охранника Френка будет небольшой, стиль вращения влево-вправо. Движение он начинает из правого нижнего угла. Затем в вечном цикле Охранник плывёт в точку (−240; −60), потом разворачивается и плывёт назад. Теперь Машинке будет непросто накормить Динозавра!



Скрипт Машинки надо немного модернизировать, чтобы она реагировала на Охранника.

В скрипт добавлено ещё одно условие, которое отправляет Машинку на старт при касании Френка.

Проект готов, сохраните его.

11.4. Задания

1. Сделайте так, чтобы Машинка издавала звуки во время движения.

2. Ускорьте движение Охранника.

3. Добавьте ещё одного Охранника.

4. Уменьшите размер героев до 20% и нарисуйте новый, более сложный лабиринт.

5. У Френка четыре костюма. Сделайте так, чтобы при движении он иногда поднимал руки вверх.

ГЛАВА 12. МУЛЬТФИЛЬМ «ЗАЯЦ И ЛИСА»

Жил да был Заяц. Бегал он по лесу и никого не трогал. Вдруг, откуда ни возьмись — Лиса Патрикеевна. Заяц попытался завести разговор о погоде, но у Лисы другие планы…

Давайте сделаем мультфильм о встрече Зайца и Лисы в районе средней полосы.

12.1. Добавляем фон и спрайты

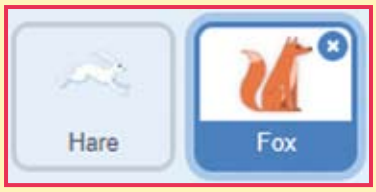
Создайте новый проект. Измените фон на лес. Удалите спрайт Кота и добавьте Зайца и Лису.

Удалите сидящий костюм у Зайца, в отличие от Лисы ему некогда сидеть на месте.

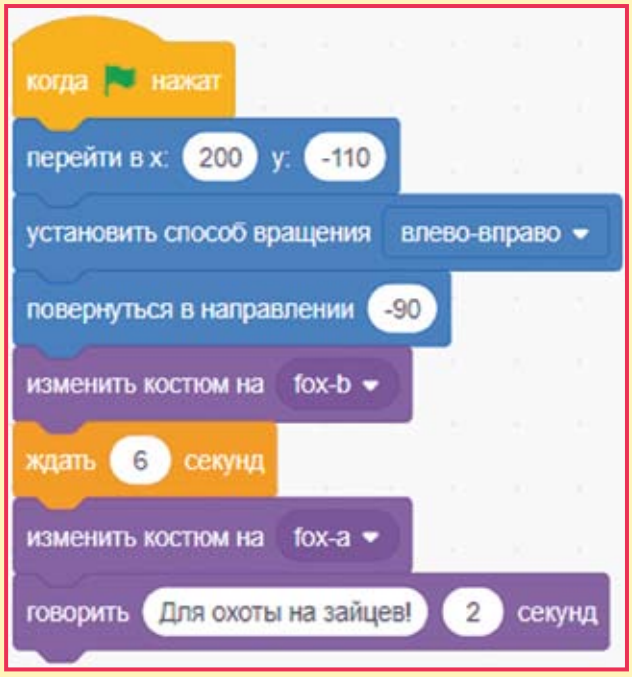
12.2. Программируем Зайца и Лису

Выберите спрайт Зайца и соберите ему вот такой скрипт.

При запуске проекта Заяц перейдёт в начальную точку, повернётся направо и изменит стиль вращения. Затем он побежит к Лисе, делая 25 раз по 10 шагов, скажет пару фраз и побежит обратно.



Выберите спрайт Лисы и соберите ей вот такой скрипт.



При запуске проекта Лиса перейдёт в начальную точку, изменит стиль вращения, повернётся влево и сменит костюм на сидящий. Затем Лиса подождёт 6 секунд, пока Заяц набегается, и выскажет ему всё, что думает о зайцах.

Проект готов, протестируйте его и не забудьте сохранить.

12.4. Задания

1. Добавьте в мультфильм музыкальное сопровождение.

2. Запишите с микрофона фразы Зайца и Лисы, добавьте их в мультфильм вместо блоков говорить.

3. Создайте новый спрайт с надписью «Конец» и запрограммируйте его на появление в конце мультфильма.

4. Сделайте свой коротенький мультфильм с двумя персонажами.

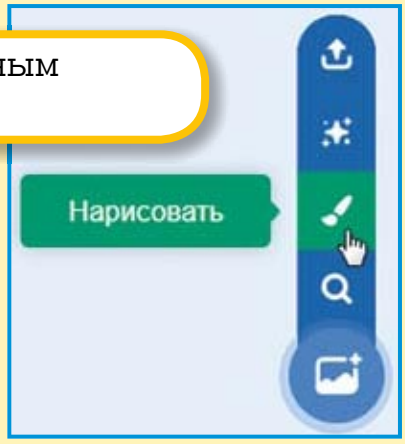
ГЛАВА 13. ИГРА «МЫШКА-НОРУШКА»

Снесла Курочка Ряба шесть простых яичек и пошла сносить седьмое — не простое, а золотое. Мимо мышка-норушка бежала, длинным хвостиком махала — кусочек вкусного сыра искала.

Сделаем игру, в которой игроку предстоит провести Мышку к заветному кусочку сыра, не разбив ни одного яичка.

13.1. Создаём спрайты и фон

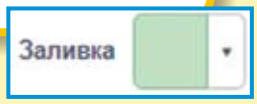
Сначала закрасим сцену светло-зелёным цветом. Нажмите кнопку **Нарисовать**.

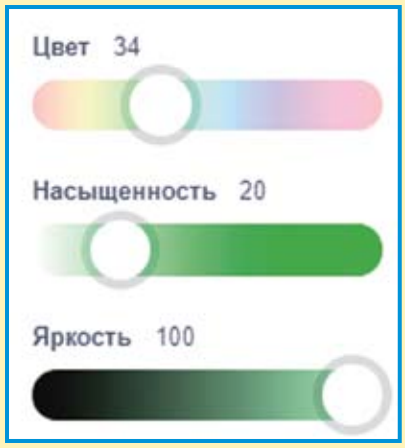


Нажмите кнопку **Конвертировать в растровую графику**.

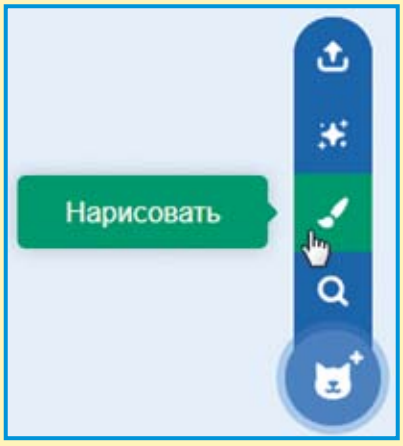
Выберите инструмент **Заливка**.

Выберите цвет 34
с насыщенностью 20
и яркостью 100.



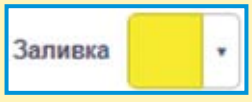


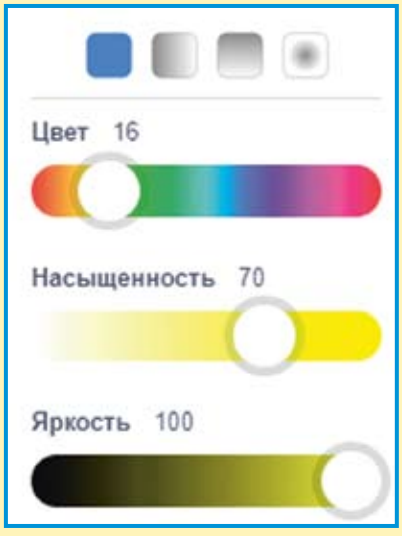
Закрасьте фон, просто щёлкнув мышью по нему.



Теперь нарисуем сыр.
Нажмите кнопку **Нарисовать**.

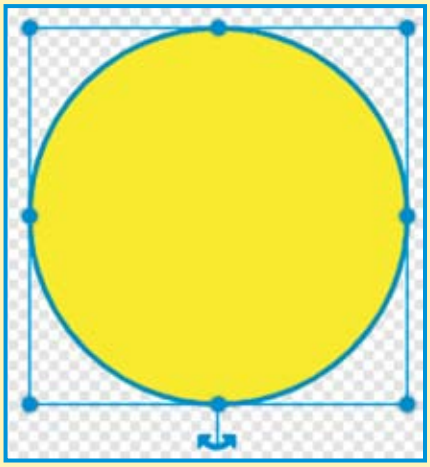
Выберите жёлтый цвет.





Выберите цвет и толщину контура

Выберите инструмент **Круг**.

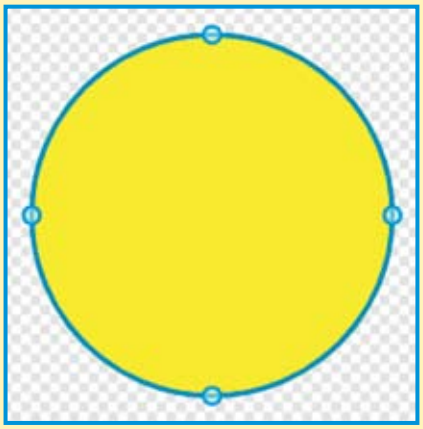


Нарисуйте жёлтый круг.

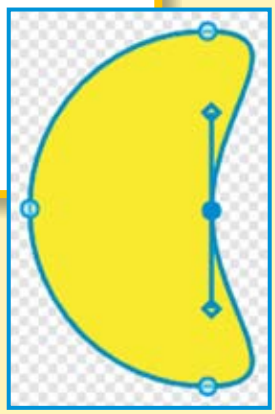
Совет

Удерживайте нажатой клавишу <Shift>, чтобы круг получился ровным.

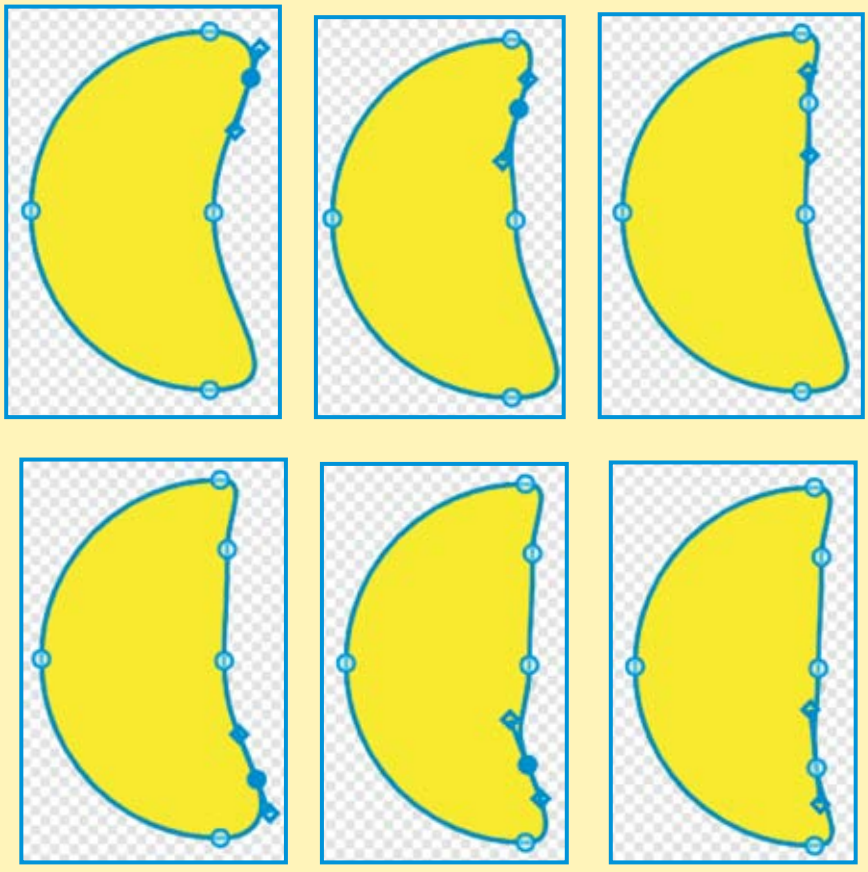
Теперь надо разрезать круг пополам с использованием инструмента **Изменение формы.**



Щёлкните по кругу — на нём появятся кругленькие точки привязки. Перемещайте их и изменяйте кривизну линий, чтобы получилась вот такая половинка круга.



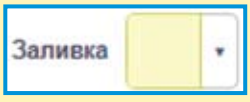
Если щёлкнуть по линии, то в этом месте появится новая точка привязки, которую можно перемещать. Если дважды щёлкнуть по существующей точке, то она пропадёт.

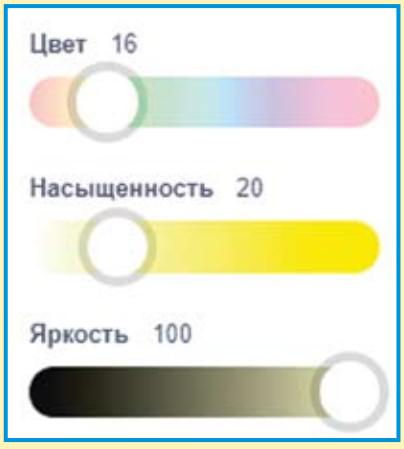


Половинка сыра готова! Теперь надо нарисовать дырки в сыре.

Выберите инструмент **Круг**.

Измените насыщенность жёлтого цвета.

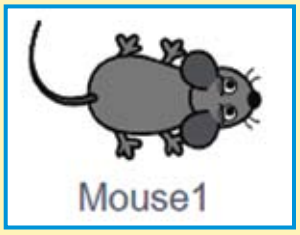




Толщину контура и цвет оставьте без изменений.

Нарисуйте дырки в сыре.

Сыр готов. Теперь добавьте из библиотеки спрайтов Мышку и Яйцо.

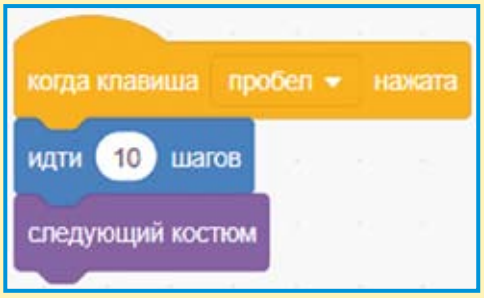


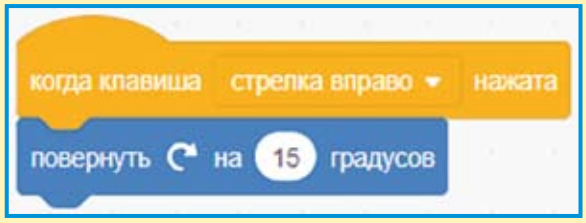
Ну вот, у нас есть закрашенный фон, Сыр, Мышка и Яйцо. Переименуйте спрайты.

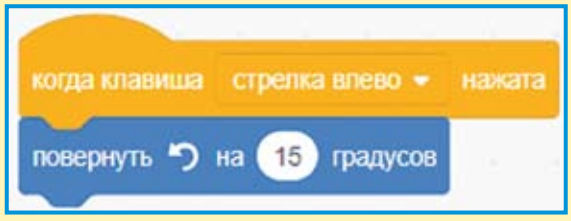
Начинаем программировать!

13.2. Программируем поведение спрайтов

Сначала запрограммируем Мышку. Соберите три скрипта для управления движением этого персонажа.

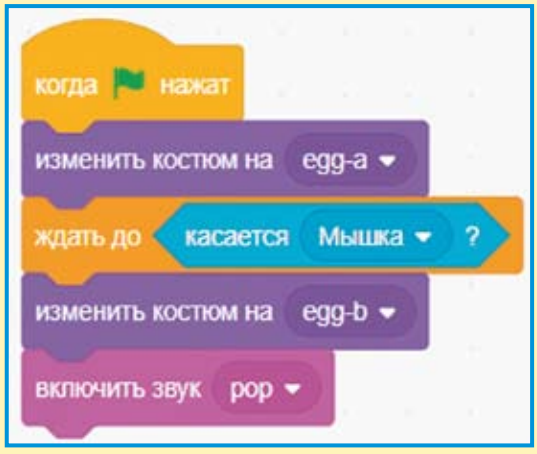






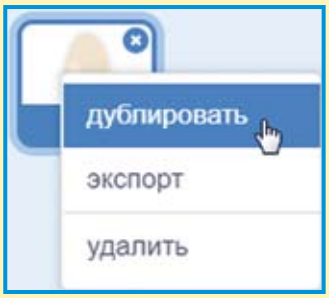
При нажатии клавиши <Пробел> Мышка будет идти вперёд, меняя костюмы, а при нажатии на клавиши-стрелочки будет поворачивать направо и налево.

Теперь надо запрограммировать Яйцо.

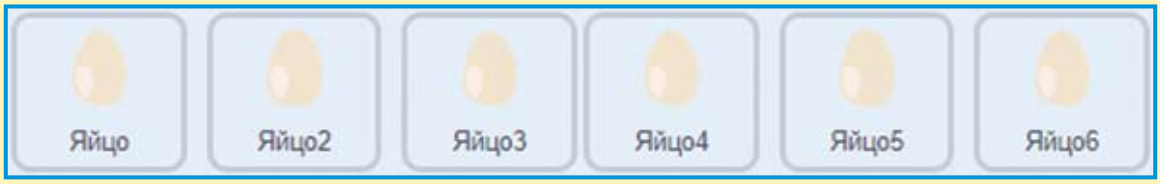


При запуске проекта Яйцо сменит костюм на целое, а затем будет ждать, пока не коснётся Мышки, и когда это произойдёт, то она сменит костюм на разбитый.

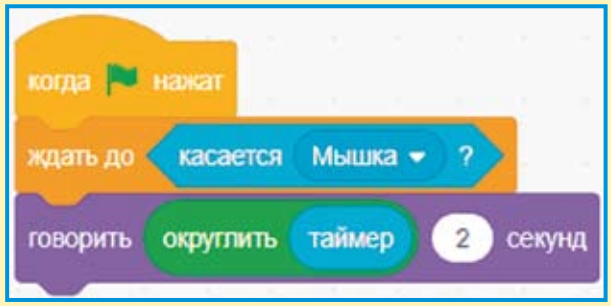
Дублируйте яйцо 5 раз.



Теперь у нас есть 6 яиц с одинаковыми скриптами!



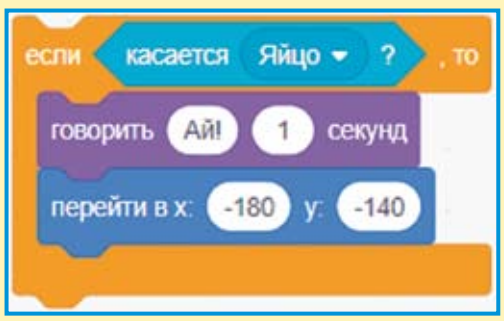
Запрограммируйте Сыр.



При запуске проекта Сыр будет постоянно ждать, пока его не коснётся Мышка, и после этого он скажет время, за которое игрок прошёл игру. Это время хранится во встроенной переменной **таймер**. Для того чтобы время было в секундах без дробной части, используйте блок округлить.

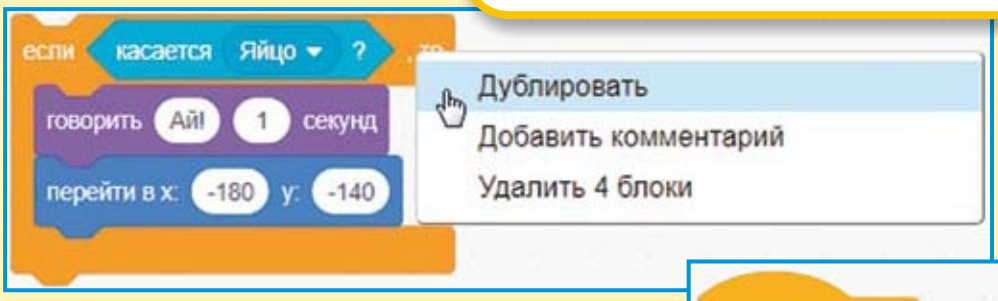
Протестируйте игру. Мышка бежит, разбивает яйца и запросто добирается до сыра. Так играть не интересно, давайте усложним игру, чтобы при касании яиц Мышка отправлялась на старт.

Соберите Мышке ещё два скрипта.



Обратите внимание! В следующих скриптах шесть раз используется повторяющаяся конструкция из блоков.

Для того чтобы проще собрать такие большие скрипты, дублируйте эту конструкцию.



При запуске проекта Мышка перейдёт на старт, затем она постоянно будет проверять, не касается ли она какого-либо яйца, и если касается, то она «айкнет» и вернётся на старт.

Этот скрипт также постоянно будет проверять, не касается ли Мышка какого-либо яйца.

Игра готова. Запустите её и попробуйте подобраться к Сыру. Сохраните игру.

13.3. Задания

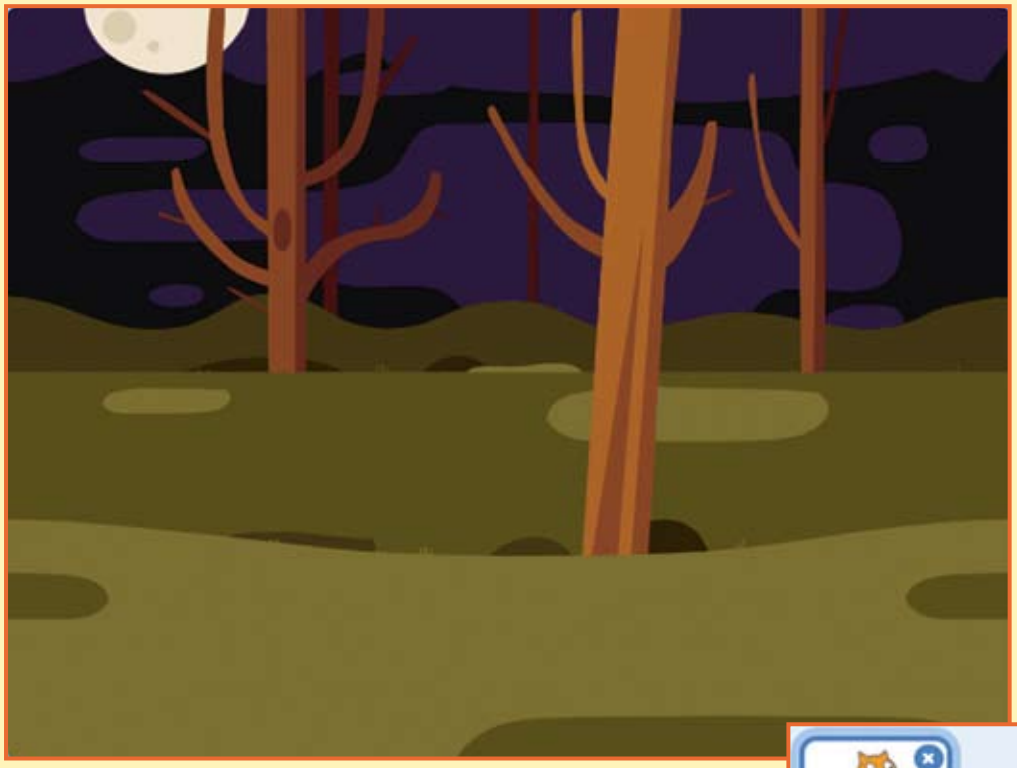
1. Увеличьте скорость движения Мышки.
2. Добавьте ещё два яйца и доработайте скрипты Мышки.
3. Сделайте так, чтобы Мышка управлялась мышью.
4. Замедлите скорость движения Мышки в 2 раза.

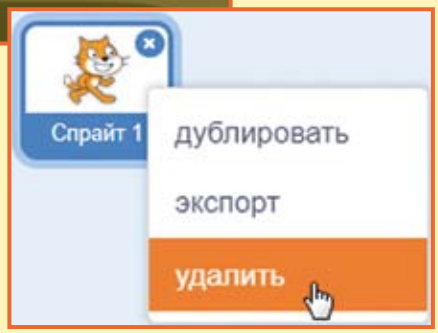
ГЛАВА 14. ИГРА «ВЕДЬМА И ВОЛШЕБНИК»

Давайте сделаем игру про магическое сражение Волшебника и Ведьмы. Всего у Волшебника есть 5 склянок с волшебными зельями.

14.1. Создаём спрайты

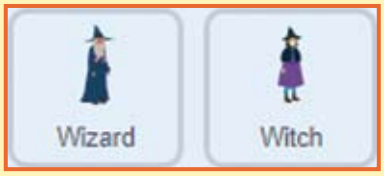
Сражение Волшебника и Ведьмы будет происходить в лесу. Выберите его из библиотеки фонов.





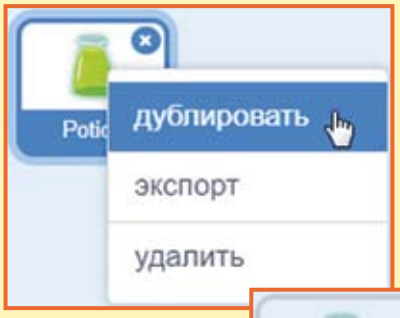
Кота в нашей истории не будет. Удалите его.

Добавьте Волшебника и Ведьму из библиотеки спрайтов.



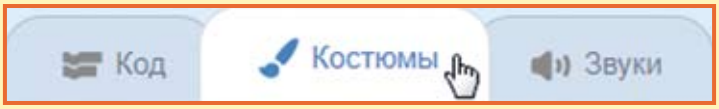
Теперь у нас есть лес и главные герои.

Добавьте из библиотеки склянку с зельями.



Дублируйте склянку пять раз.

Перейдите на вкладку **Костюмы**, переключите костюмы склянок и перекрасьте их.

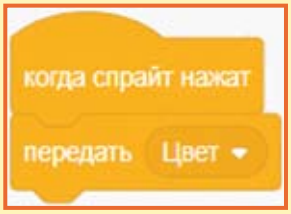


Разместите арсенал Волшебника в левой части экрана.

Все спрайты готовы, можно начинать программировать!

14.2. Программируем спрайты

Сделайте вот такую программу для зелёной склянки.



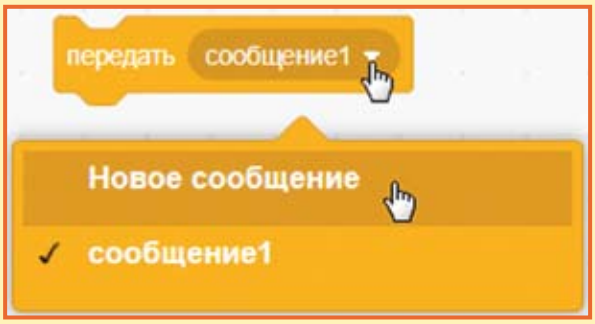
Здесь использованы сразу два новых блока: когда спрайт нажат и передать. Как вы, наверное, догадались, блок когда спрайт нажат позволяет выполнить то, что прицеплено к нему в том случае, когда щёлкнули по спрайту.

Блок передать осуществляет передачу сообщения с указанным именем.

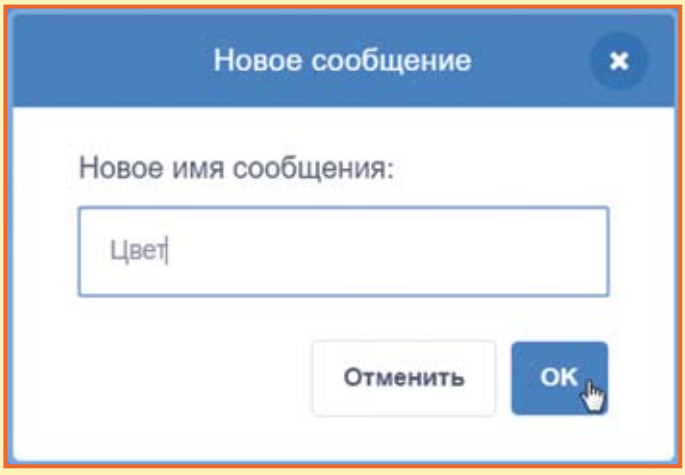
В нашем случае при щелчке на зелёной склянке будет передано сообщение с именем «Цвет». Сообщения нужны для того, чтобы остальные спрайты проекта понимали, что происходит, и реагировали, как задумано. В нашем случае на сообщения будут реагировать Ведьма и Волшебник. Волшебник будет произносить фразу, а Ведьма станет магически видоизменяться.

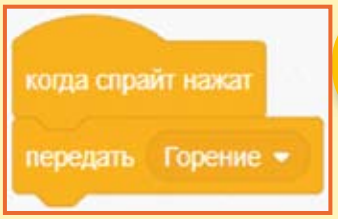
Совет

Для того чтобы ввести новое сообщение, необходимо раскрыть выпадающий список и выбрать команду **Новое сообщение**.



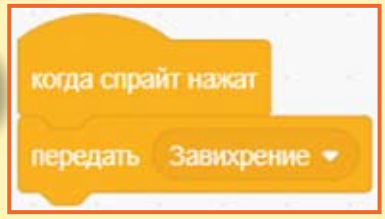
В окне ввода надо ввести имя сообщения и нажать кнопку **OK**.

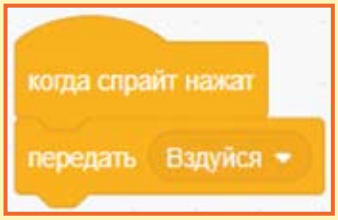




Для синей склянки сделайте вот такую программу.

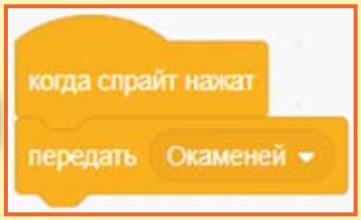
Для фиолетовой склянки вот такую.

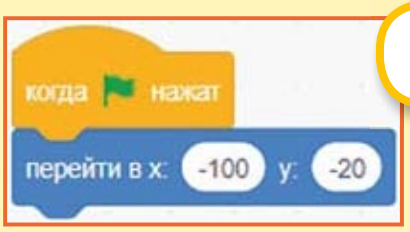




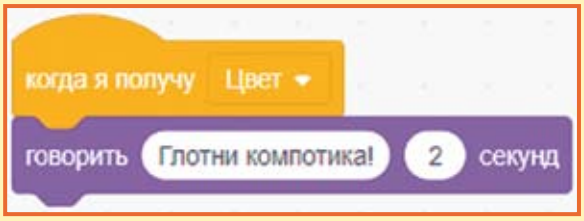
Для розовой вот такую.

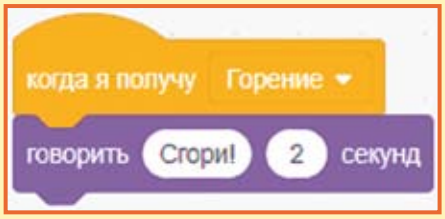
Для оранжевой вот такую.

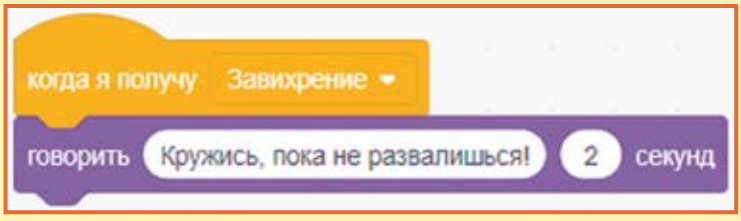


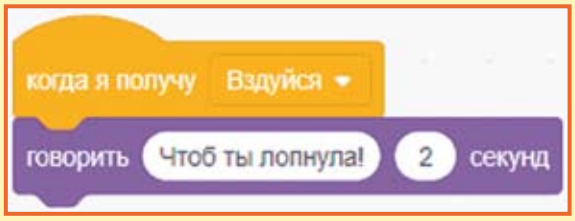


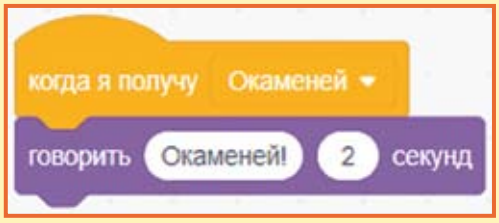
Теперь запрограммируйте Волшебника. У него будет 6 скриптов.

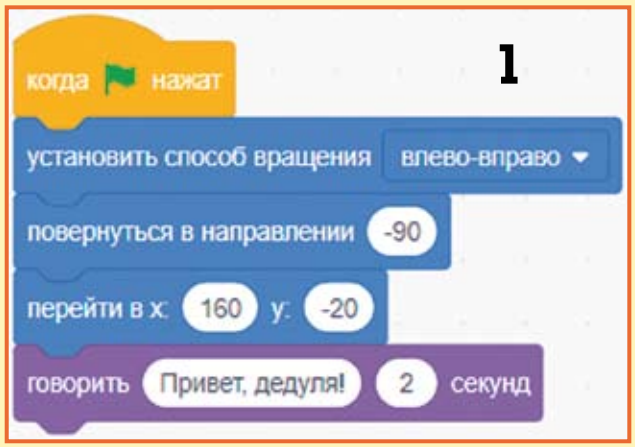


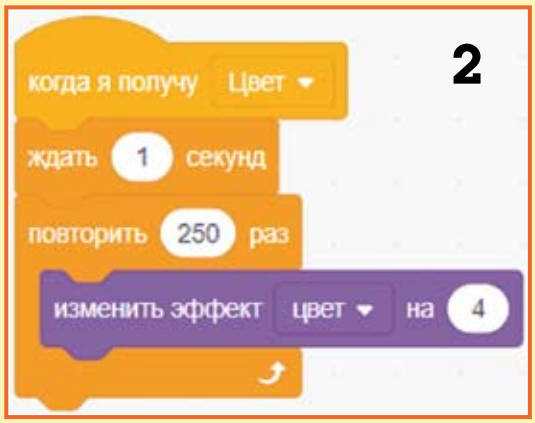


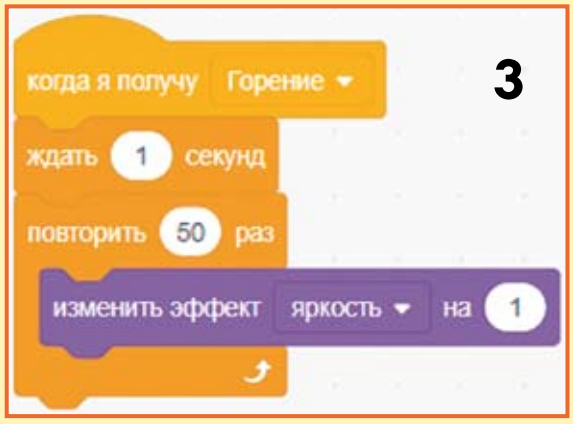


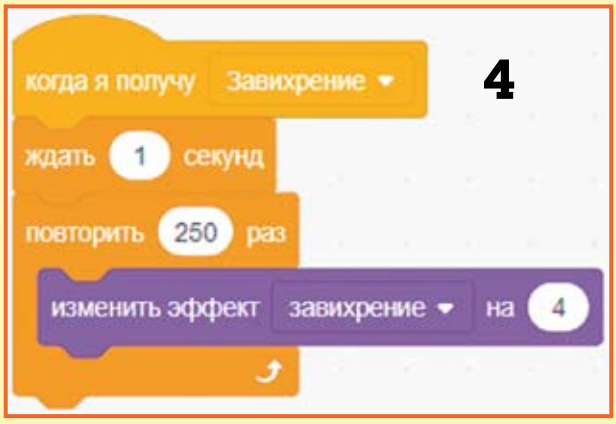




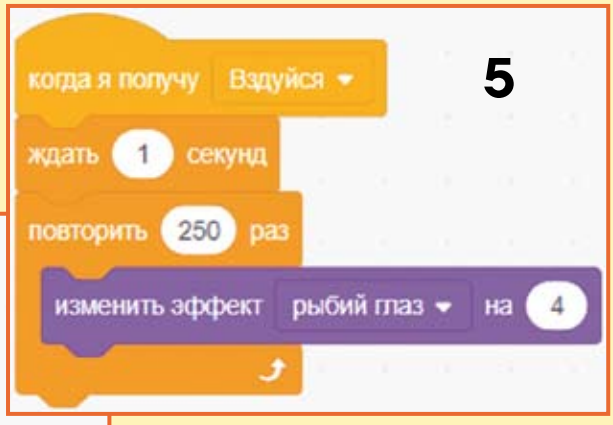


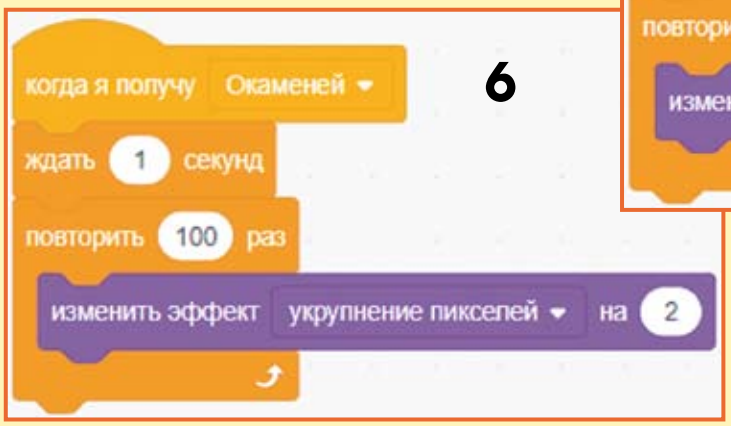






Соберите программу для Ведьмы. Она состоит из шести скриптов.



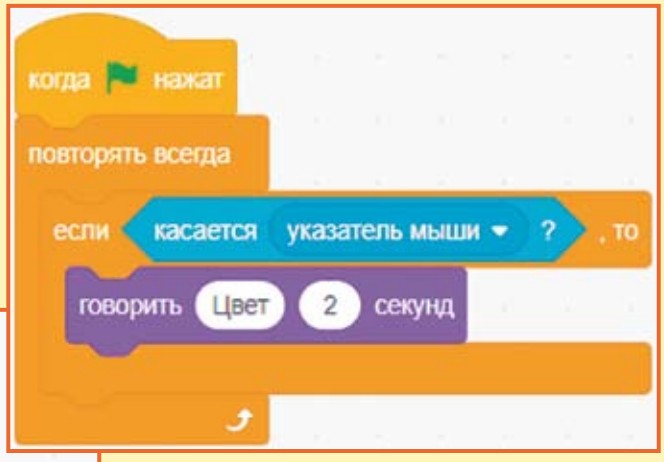


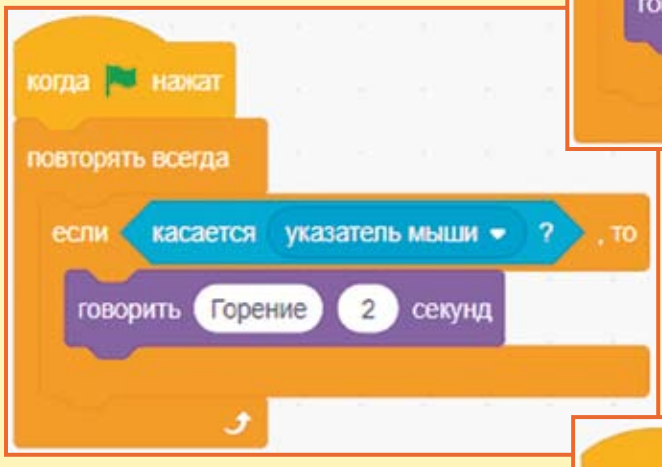
При получении различных сообщений Ведьма будет изменять внешний вид в соответствии с заклинанием.

14.3. Всплывающие подсказки

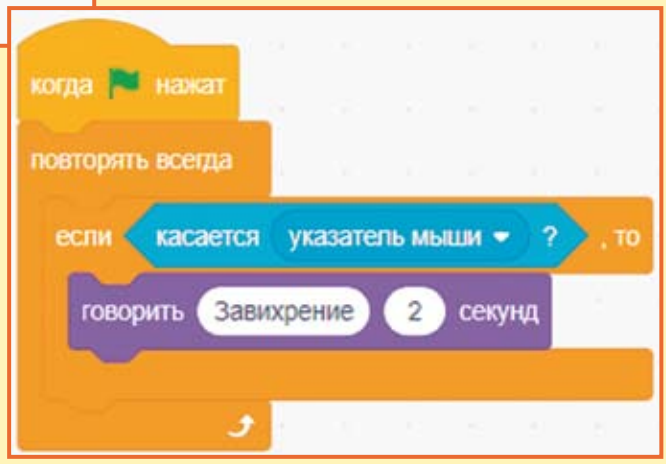
Я думаю, будет хорошей идеей добавить каждой склянке всплывающую подсказку, чтобы при наведении на неё курсора мыши склянка сообщала нам о своём содержимом. Так Волшебнику будет проще расправиться с противником.

Добавьте зелёной склянке вот такой скрипт.



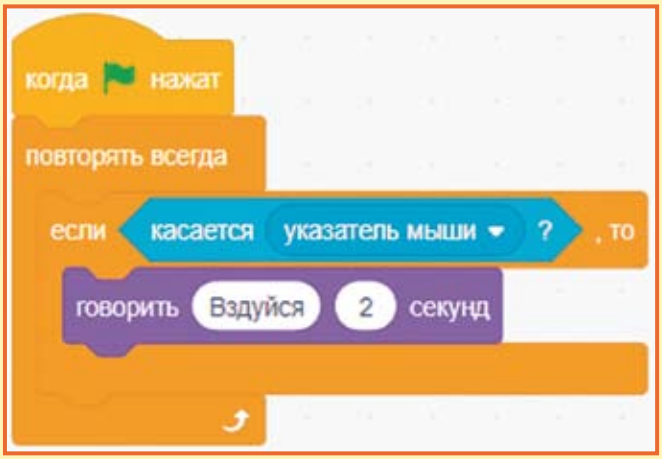


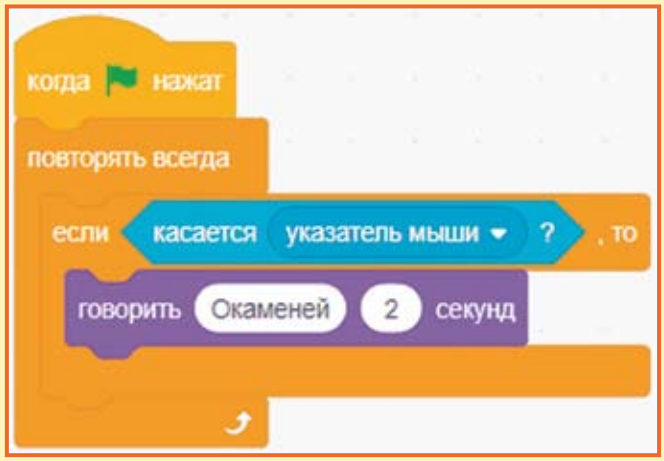
Добавьте синей склянке вот такой скрипт.



Добавьте фиолетовой склянке вот такой скрипт.

Добавьте розовой склянке вот такой скрипт.





Добавьте оранжевой склянке вот такой скрипт.

Проект готов, протестируйте его и сохраните.

14.4. Задания

1. Сделайте новое зелье, которое будет накладывать на Ведьму сразу несколько эффектов.

2. Сделайте несколько зелий для Ведьмы, чтобы она могла отбиваться от Волшебника.

3. Сделайте так, чтобы при нажатии на склянку она на некоторое время становилась ярче, а потом снова возвращалась в обычное состояние.

ГЛАВА 15. КОТ-МАТЕМАТИК

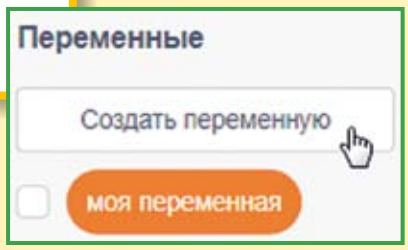
15.1. Переменные

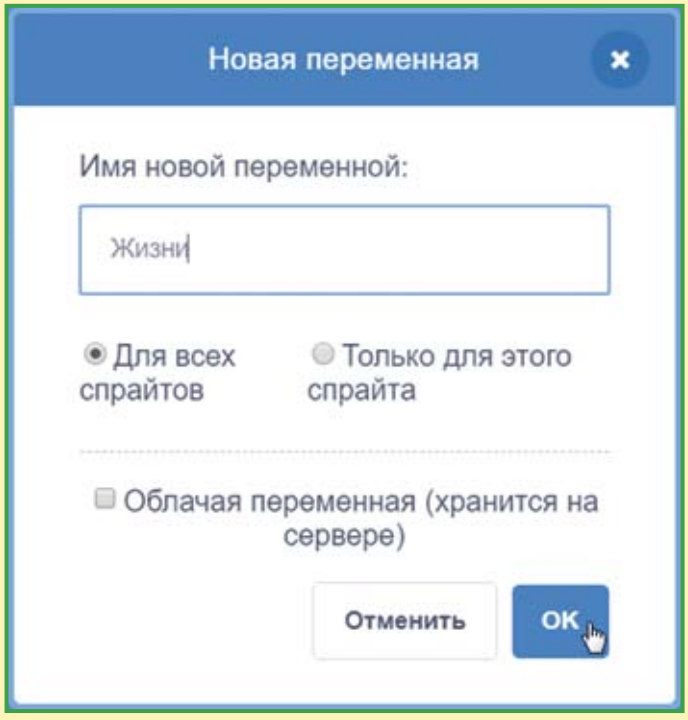
Жил да был очень умный Котик. Он жил на рынке рядом с прилавком, на котором продавали рыбу. Весь день он наблюдал за торговлей и научился считать.

В этой главе мы сделаем проект про Кота-математика. Кот будет очень грамотным и сможет складывать числа.

При создании этого проекта нам понадобятся переменные. *Переменные* в Scratch — это такие овальчики, которые хранят в себе некоторые значения.

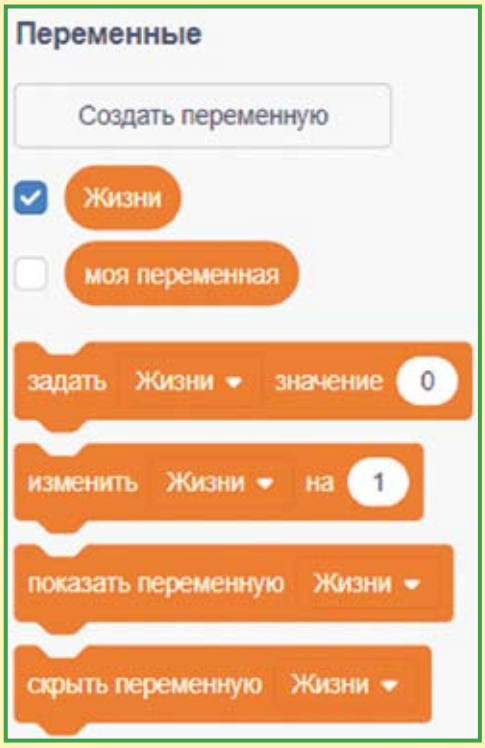
Для того чтобы создать переменную, нужно выбрать оранжевые блоки **Переменные** и нажать кнопку **Создать переменную**.





Необходимо ввести имя переменной, например **Жизни**, и нажать кнопку **OK**.

Появится новая овальная переменная.

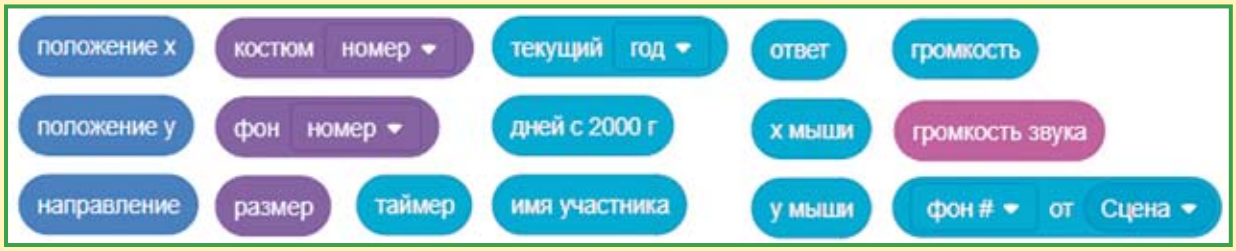


Блок задать значение позволяет записать в переменную значение. Оно может быть числом или словом. Если переменная хранит числовое значение, то блок изменить может изменить значение переменной. Блоки показать переменную и скрыть переменную делают то же самое, что и маленькая галочка рядом с именем переменной — отображают или скрывают значение переменной на сцене.

Это выглядит вот так.

В Scratch есть два вида переменных: пользовательские и защищённые. *Пользовательские переменные* оранжевого цвета, вы можете создавать их, удалять, можете присваивать им любые значения. *Защищённые переменные* — это переменные разных

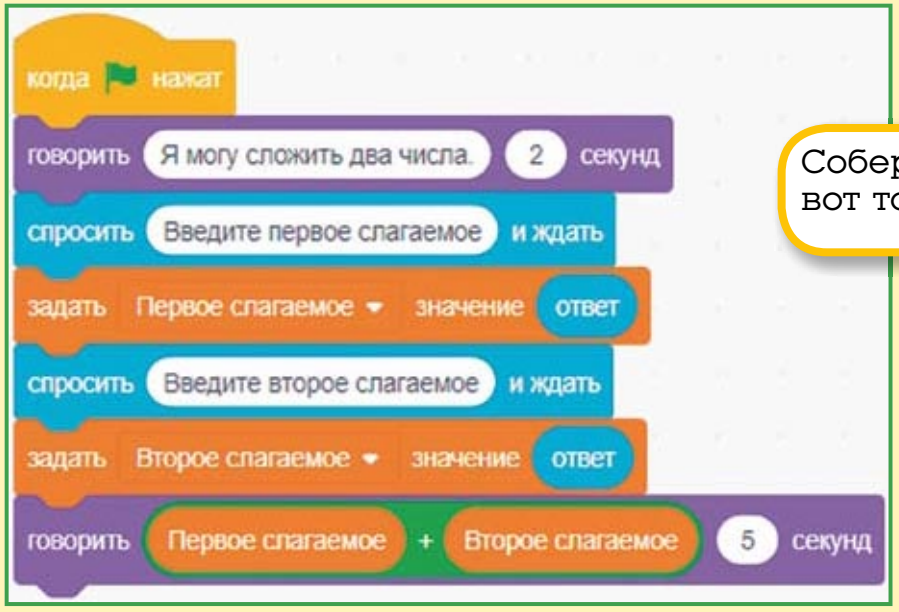
цветов, изменить значение которых вы не можете. Эти переменные хранят информацию, которую получают автоматически. Вот все защищённые переменные Scratch.



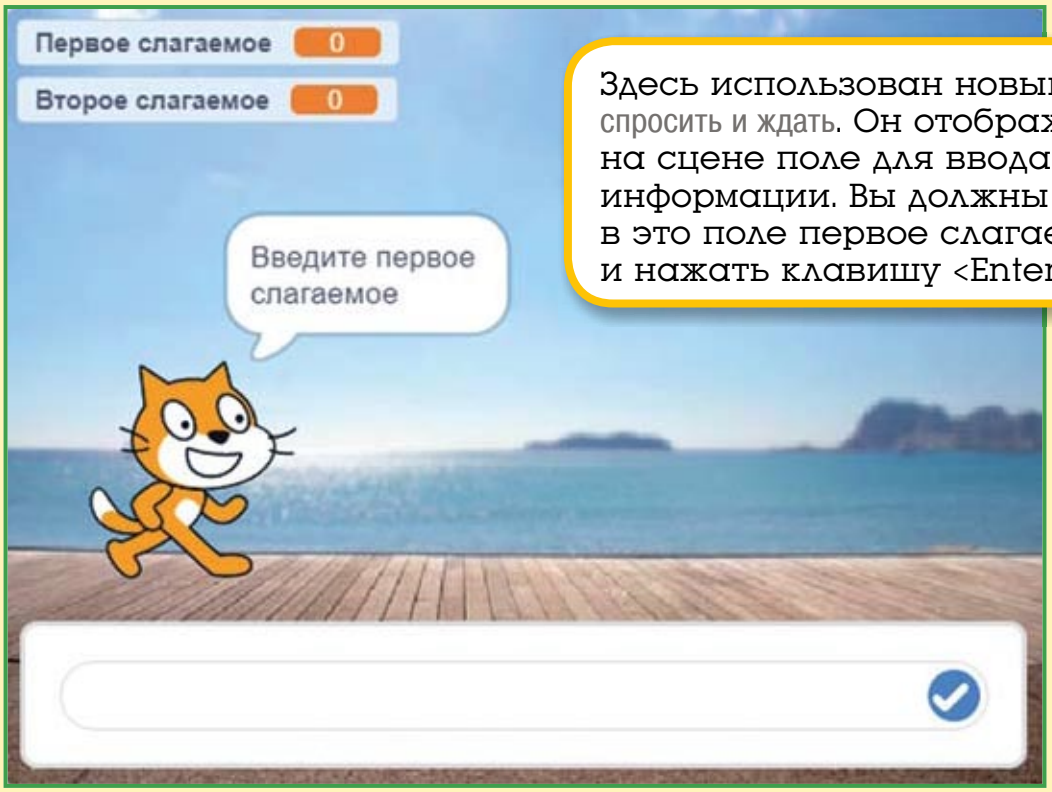
Синие защищённые переменные хранят информацию о координатах спрайта и его направлении; фиолетовые — порядковый номер костюма, имя фона и размер спрайта; сиреневая переменная хранит громкость звука; голубые — ответы, которые вы будете вводить, время, положение курсора, другие параметры спрайтов и сцены.

15.2. Программируем игру

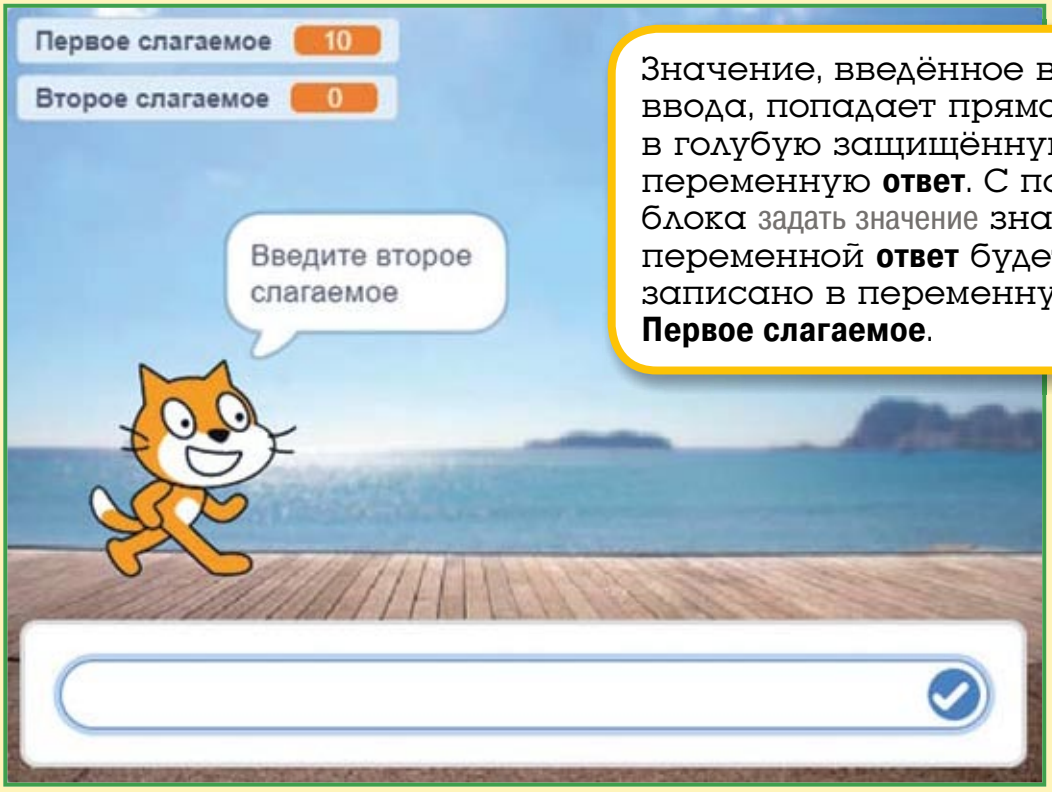
Теперь, когда вы познакомились с переменными, можно сделать игру про Кота-математика. Сначала создайте две переменные: **Первое слагаемое** и **Второе слагаемое**.



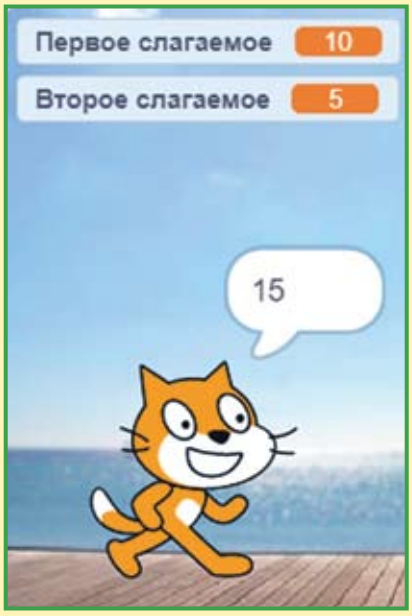
Соберите для Кота вот такой скрипт.



Здесь использован новый блок спросить и ждать. Он отображает на сцене поле для ввода информации. Вы должны ввести в это поле первое слагаемое и нажать клавишу <Enter>.



Значение, введённое в поле ввода, попадает прямо в голубую защищённую переменную **ответ**. С помощью блока задать значение значение переменной **ответ** будет записано в переменную **Первое слагаемое**.



Следующее значение, введённое в поле ввода, попадёт в переменную **ответ**, и сразу после этого будет записано в переменную **Второе слагаемое**.

Котик сообщит сумму слагаемых. Какой молодец! Для получения ответа он использовал оператор сложения чисел.

В каждое из полей этого блока вставляется какая-либо переменная. В нашем случае в поля вставлены слагаемые. С помощью блока говорить Кот произносит сумму слагаемых.

Сохраните проект.

15.3. Задания

1. Измените проект так, чтобы Кот умножал введённые значения.

2. Измените проект так, чтобы Кот находил сумму трёх чисел.

3. Добавьте в проект банкира, который будет переводить введённую сумму в рублях в сольдо по курсу 275 рублей за сольдо. (Надеюсь, вы читали повесть-сказку Алексея Николаевича Толстого «Золотой ключик, или Приключения Буратино»? У Буратино там были сольдо.)

4. Добавьте в проект заправщика, который будет спрашивать, сколько литров бензина залить, и говорить стоимость при цене 50 рублей за литр бензина.

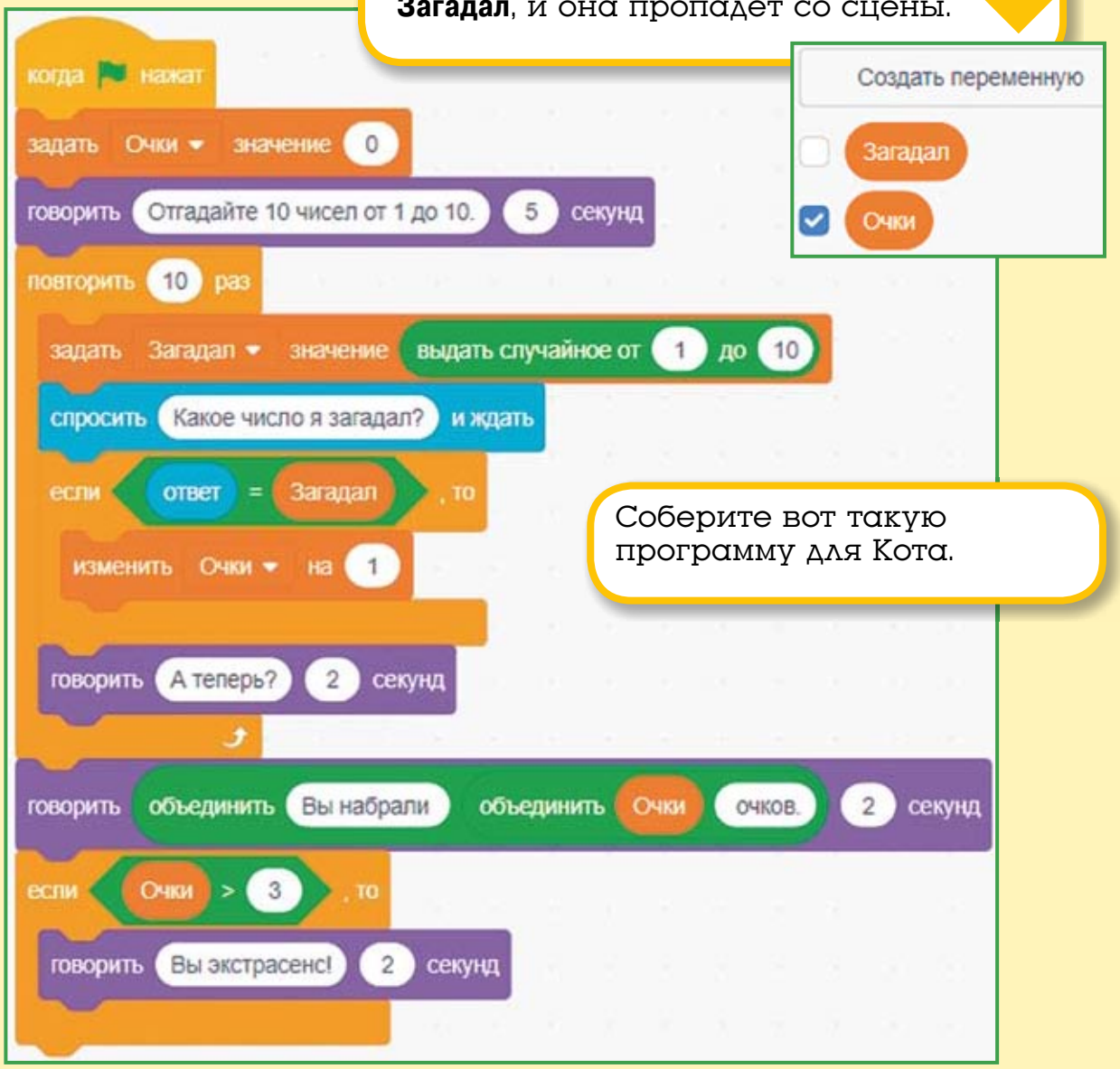
5. Добавьте в проект автомобиль, который потребляет 10 литров бензина на 100 километров пути. Пускай этот автомобиль спрашивает расстояние, которое вы собираетесь проехать, и сообщает о затратах на бензин при цене 50 рублей за литр.

15.4. Отгадай число

Давайте сделаем ещё один математический проект — игру про отгадывание чисел. Кот будет загадывать число от 1 до 10, а задача игрока — угадать, какое число загадано. Всего 10 попыток.

Создайте новый проект и добавьте в него две переменные: **Очки** и **Загадал**.

Снимите галочку с переменной **Загадал**, и она пропадёт со сцены.



Соберите вот такую программу для Кота.

В начале работы программы переменная **Очки** будет обнулена. Затем Кот скажет: «Отгадайте 10 чисел от 1 до 10». Потом 10 раз повторится цикл загадки. Сначала переменной **Загадал** будет задано случайное значение с помощью блока выдать случайное. Случайное значение будет выбрано из диапазона от 1 до 10. Затем Кот спросит: «Какое число я загадал?» Ваш ответ будет помещён в защищённую переменную **Ответ**. Блок если сравнит переменные **Ответ** и **Загадал**. Если они равны, то переменная **Очки** увеличится на единицу, и Кот предложит отгадывать следующее число. После 10 проходов цикла Кот скажет, сколько очков вы набрали, и если вы набрали больше 3 очков, то вы экстрасенс! У меня не получается угадать больше одного числа.

Сохраните проект.

15.5. Задания

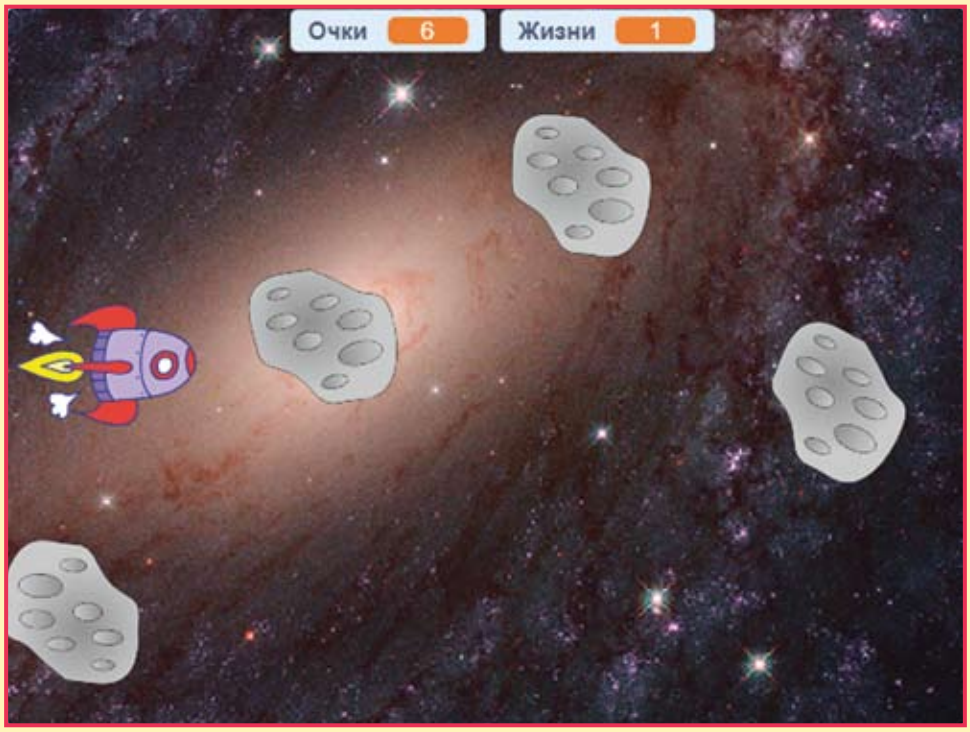
1. Добавьте ещё одну переменную — **Номер попытки**. Пусть в ней отображается значение текущей попытки.

2. Сыграйте в игру 10 раз и запишите набранные очки на бумаге. Измените диапазон случайных чисел от 1 до 5. Снова запишите ответы на бумаге. Почему теперь вы можете набрать больше очков?

3. Сделайте так, чтобы Кот спрашивал не случайное число, а примеры из таблицы умножения. Для этого вам понадобятся ещё две переменные: **Слагаемое1** и **Слагаемое2**.

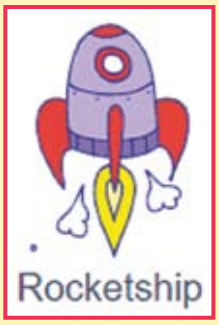
ГЛАВА 16. ИГРА «КОСМИЧЕСКИЙ ПОЛЁТ»

Смелый Кот залез в ракету и отправился в космос. Навстречу ему летят космические камни (метеороиды). Игроку надо управлять ракетой с помощью мыши и аккуратно облететь опасные препятствия, ведь у него всего 3 жизни! Цель игры — набрать 50 очков.



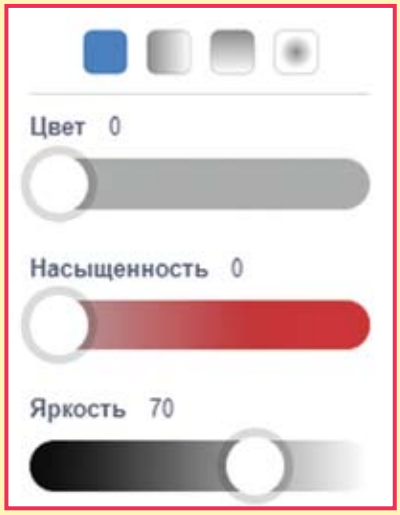
16.1. Создаём спрайты и фон

Так как дело происходит в космосе, то добавьте космический фон.



Спрайт Кота удалите. Добавьте спрайт Ракеты.

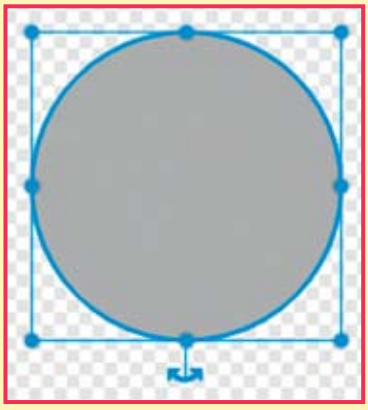
Нарисуйте спрайт метеороида. Выберите инструмент **Круг**.



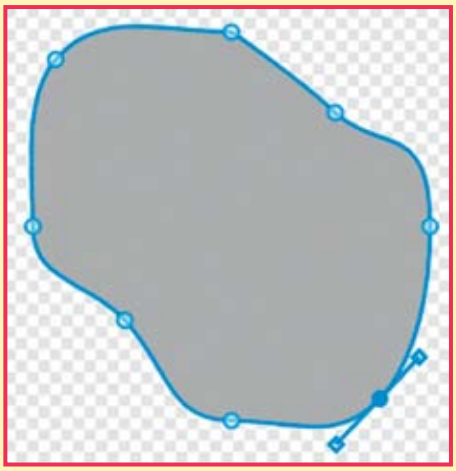
Выберите серый цвет.

Выберите чёрную границу толщиной 1 пиксел.

Нарисуйте серый круг.



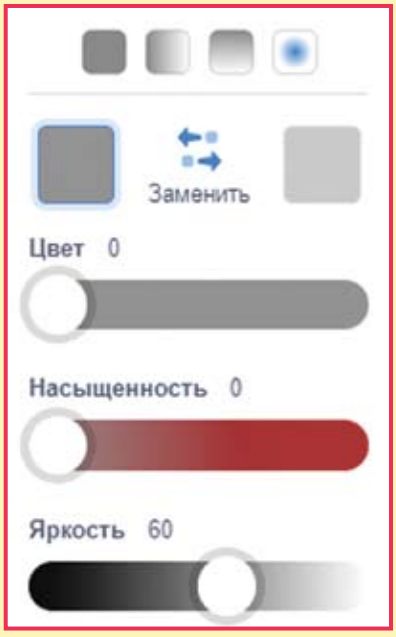
Выберите инструмент изменения формы.



Измените форму космического камня.

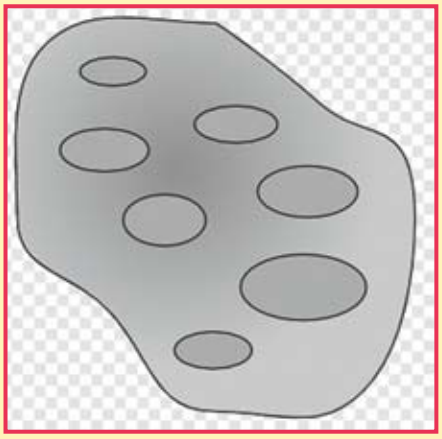
Теперь нарисуйте на нём маленькие кратеры с помощью инструмента **Круг**.

Теперь выберите инструмент **Заливка**.



Выберите два оттенка серого цвета и радиальный способ заливки.

Перекрасьте метеороид, это придаст ему объём.

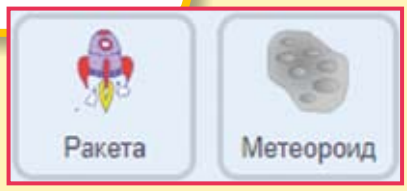


Теперь выберите градиентный тип заливки и перекрасьте кратеры.

Теперь они стали похожи на углубления в космическом камне.

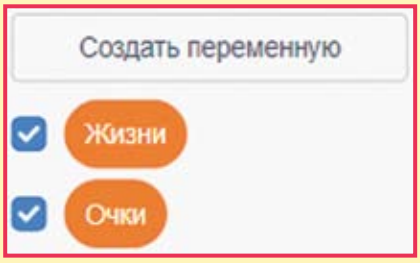
Спрайт космического метеороида готов.

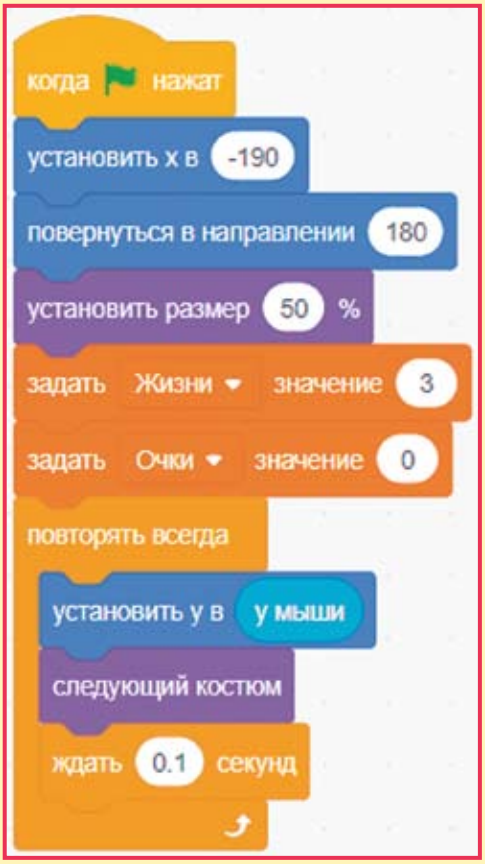
Переименуйте спрайты.



16.2. Программируем спрайты

Сначала создайте две переменные: **Очки** и **Жизни**. Переменная **Очки** будет вести подсчёт очков до достижения 50, а переменная **Жизни** будет показывать оставшиеся жизни Ракеты.





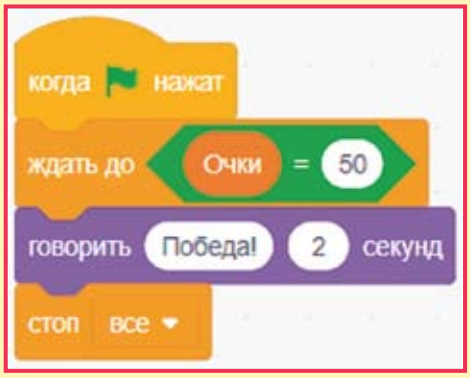
Сделайте 3 скрипта для Ракеты. Первый скрипт такой.

Первый скрипт перемещает Ракету в левую часть сцены, поворачивает её направо и задаёт размер в два раза меньше обычного. Далее задаёт начальные значения переменных, а затем постоянно устанавливает координату *Y* Ракеты в координату *Y* мыши. В результате вертикальное положение Ракеты задаётся движением мыши. В горизонтальном направлении Ракета не двигается.

Эксперимент

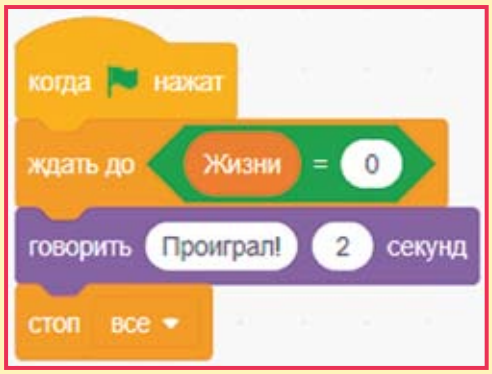
Удалите блок ждать и посмотрите, как будет вращаться Ракета.

Теперь второй скрипт.



Это скрипт победы. Блок ждать до будет постоянно ждать, когда значение переменной **Очки** станет равно 50. Когда это случится, Ракета закричит: «Победа!»

Третий скрипт такой.

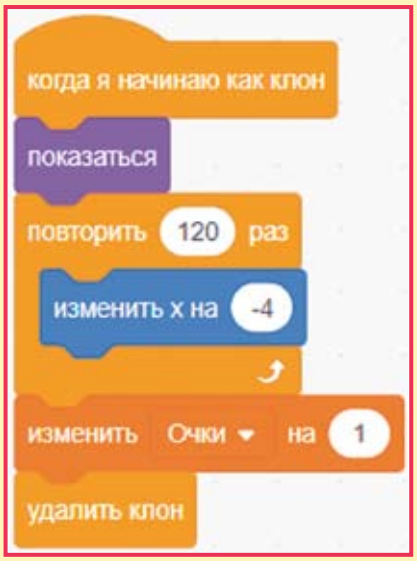


Третий скрипт Ракеты ждёт, пока значение переменной **Жизни** не станет равно нулю. Когда это произойдёт, Ракета сообщит о проигрыше.

Теперь запрограммируйте космический камень, у него 3 скрипта.

Первый скрипт такой.

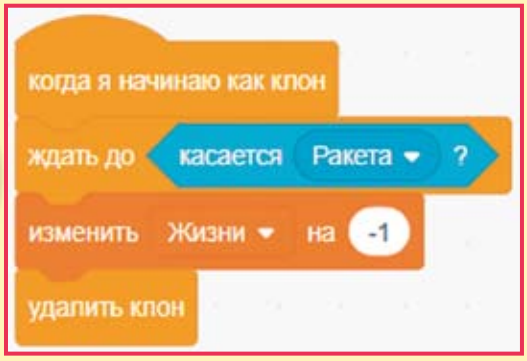
Первый скрипт Метеороида прячет его, уменьшает в 2 раза, переносит к правой границе сцены, а затем создаёт 55 клонов с интервалом в 1 секунду.



Второй скрипт такой.

Второй скрипт управляет движением клона Метеороида налево. Он 120 раз изменяет координату X на -4, затем увеличивает переменную **Очки** и удаляет клон.

Теперь соберите третий скрипт.



Этот скрипт всегда ждёт, пока Метеороид не коснётся Ракеты. Когда это произойдёт, переменная **Жизнь** уменьшится на 1. Протестируйте игру и сохраните её.

16.3. Задания

1. Сделайте так, чтобы при касании Метеороида Ракета на 0.1 секунды изменяла свою яркость.
2. Добавьте Ракете ещё две жизни.
3. Ускорьте полёт Метеороидов.
4. Добавьте в проект частицы космической пыли, которые летят, не причиняя вреда Ракете.

ГЛАВА 17. ПОЛЁТ С УСКОРЕНИЕМ — ФЛЭППИ БЁРД

17.1. Создаём спрайты и фон

Давайте сделаем свою версию популярной игры «Флэппи Бёрд». Как вы помните, задача птички — пролететь между труб и набрать как можно больше очков.

Сначала нарисуйте фон. Он может быть любым. Я нарисовал лето, поле с цветами и облака. Вы можете поместить Птичку в более экстремальные условия. Включите фантазию!

Для запуска игры мы будем использовать спрайт Play (произносится как «плэй») в виде кнопки. Создайте новый спрайт с помощью кнопки **Нарисовать.** Нарисуйте кнопку в виде синего прямоугольника.

Затем выберите инструмент **Текст**.

Напишите на кнопке слово «Play».

Расположите кнопку в центре сцены.

Теперь нарисуйте Птичку. Это новый спрайт, создайте его, нажав кнопку **Нарисовать**. Затем переключитесь в режим растровой графики, нажав кнопку **Конвертировать в растровую графику**.

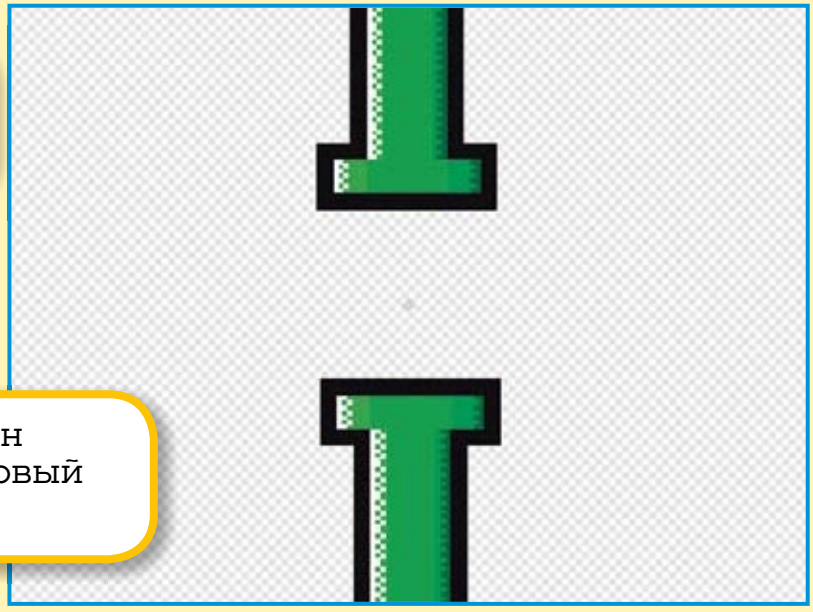
У Птички должно быть 2 костюма для анимации крыльев.

Вот первый костюм.

Вот второй костюм.

Как видите, они отличаются только формой крыльев.

Теперь создайте Трубы. У них будет 5 разных костюмов.

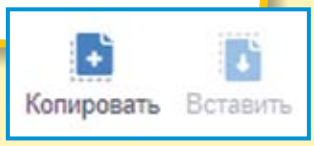


Вот так должен выглядеть первый костюм.

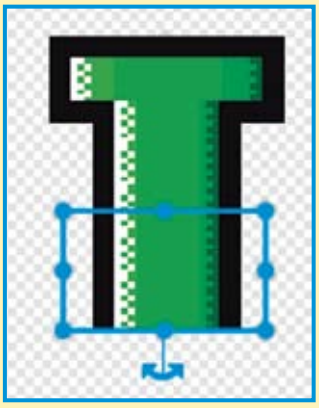
Для создания следующих костюмов дублируйте первый костюм и просто переместите трубы выше или ниже.

Для перемещения используйте инструмент **Выбрать**.

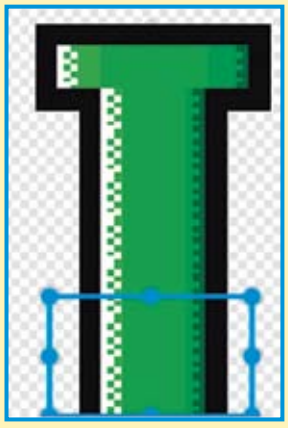
Для того чтобы удлинить часть трубы, используйте инструменты **Копировать** и **Вставить**.



Выделите часть трубы и нажмите на кнопку **Копировать**.



Вставьте скопированное и переместите на новое место.



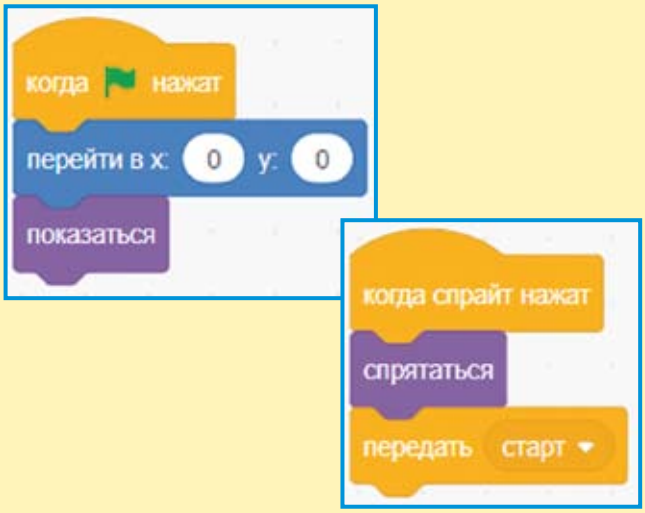
Создайте пять вот таких костюмов.

Все спрайты готовы, переименуйте их.

Теперь можно начинать программировать!

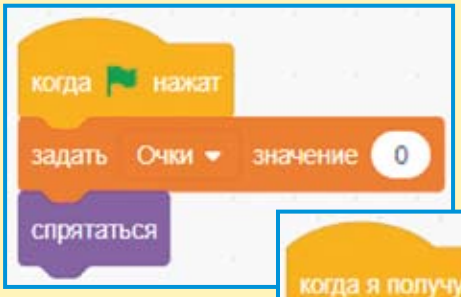
17.2. Программируем поведение спрайтов

Сначала запрограммируйте Кнопку.

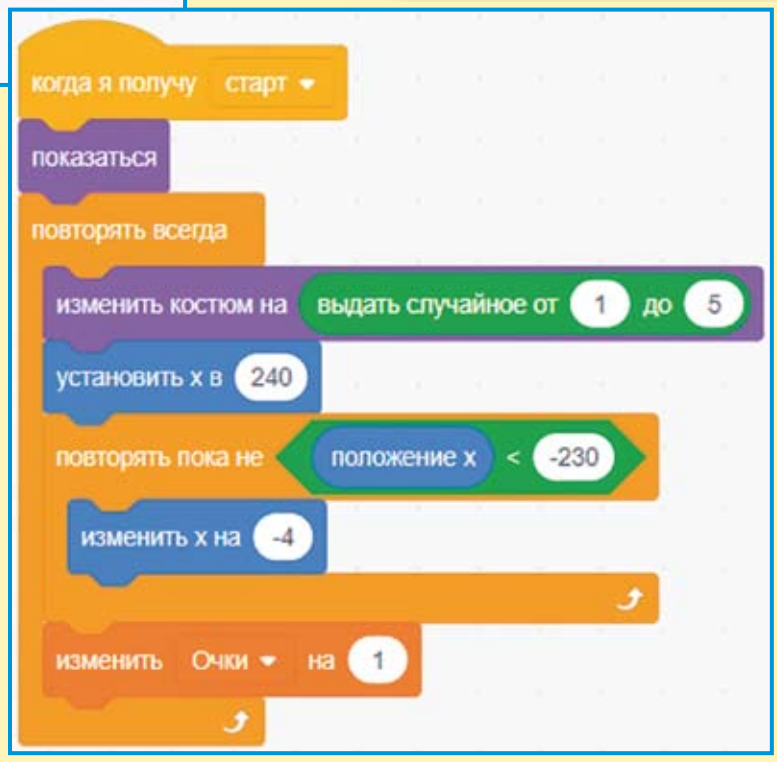


У неё очень простая программа. При запуске игры она появится в центре экрана в точке (0; 0), а затем при нажатии на неё спрячется и передаст сообщение «старт».

Теперь сделайте вот такую программу для Труб.



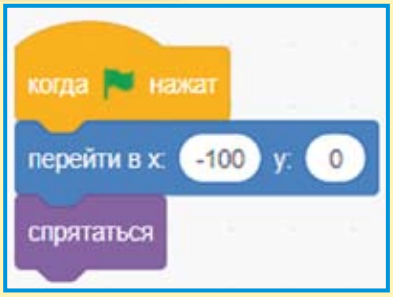
Сначала создайте переменную **Очки**, она будет хранить количество очков, набранное игроком.

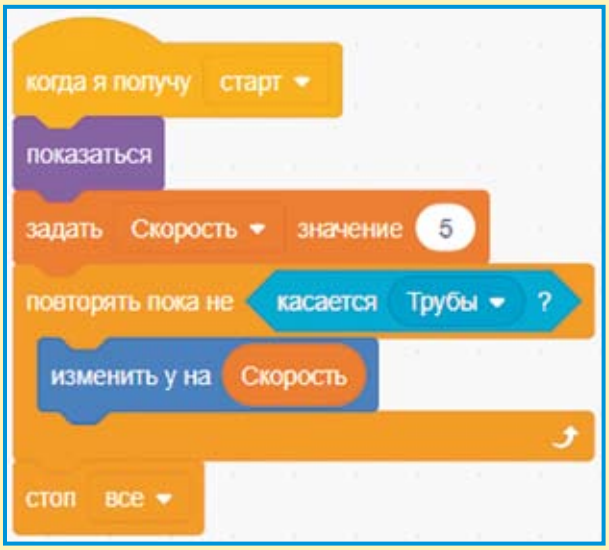


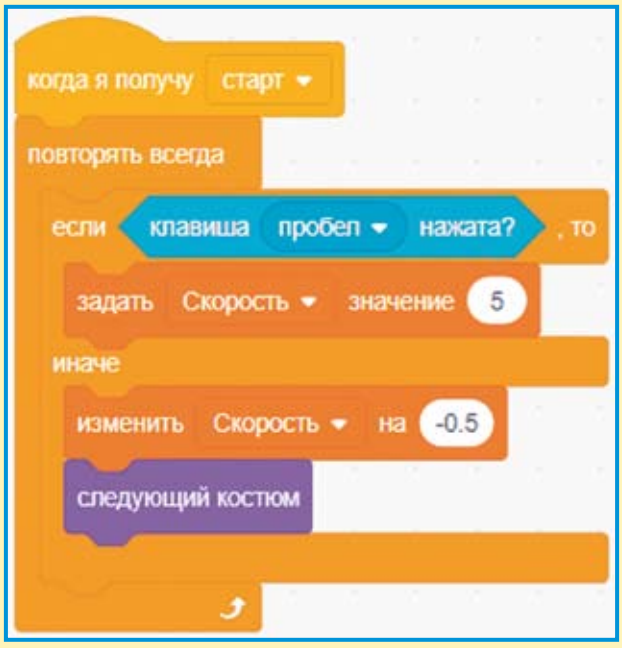
Основной скрипт Труб запускается после получения сообщения «старт». Сначала Трубы изменяют свой костюм на один из пяти, случайно выбранный с помощью блока выдать случайное. Потом Трубы перемещаются на правую границу сцены, где $X = 240$. Затем Трубы медленно двигаются налево, в цикле изменяя свою координату X на -4. Здесь использован новый блок цикла с условием повторять пока не. Этот блок повторяет своё содержимое до тех пор, пока не будет выполнено условие. В нашем случае он будет перемещать Трубы влево до тех пор, пока они не окажутся на левом краю сцены в точке, где X меньше -230. После того как Трубы «доплывут» до левой границы сцены, выполнение цикла будет завершено, и переменная **Очки** увеличит своё значение на 1.

Теперь соберите скрипты для Птички.

Сначала создайте ещё одну переменную — **скорость**. Она будет отвечать за скорость вертикального перемещения Птички (то есть вверх и вниз). Вправо и влево Птичка перемещаться не будет.







В начале работы программы Птичка перемещается в точку (−100; 0) и ждёт, когда игрок нажмёт кнопку **Play**. После нажатия **Play** и получения сообщения «старт» Птичка всегда будет изменять координату *Y* на величину, которая хранится в переменной **Скорость**. В начале работы программы **Скорость** = 5. Это значит, что Птичка будет лететь вверх, изменяя *Y* на 5. В этом же цикле постоянно проверяется условие, не коснулась ли Птичка Труб? Если это произойдёт, то работа программы будет тут же прекращена.

Третий скрипт Птички отвечает за изменение её вертикальной скорости — за «физику» игры. Вы, конечно, знаете, что любой предмет, брошенный вверх, летит с замедлением. Вертикальная

скорость брошенного предмета постоянно уменьшается до тех пор, пока в верхней точке полёта не станет равной нулю — предмет перестанет подниматься и начнёт падать. Это означает, что скорость подъёма стала отрицательной, и направление движения изменилось с подъёма на падение. Для того чтобы смоделировать этот закон физики в игре, мы будем постоянно уменьшать значение переменной **Скорость** на 0.5. При нажатии клавиши <Пробел> Птичка будет взлетать и устанавливать значение переменной **Скорость** равным 5. Как только клавиша <Пробел> будет отпущена, Птичка перестанет махать крыльями и продолжит уменьшать **Скорость** на 0.5, меняя костюмы (махая крыльями).

Игра готова, попробуйте набрать как можно больше очков.

Эксперимент

Изменяйте начальное значение переменной **Скорость** и величину её отрицательного изменения. Посмотрите, как будет меняться полёт Птички.

Сохраните проект.

17.3. Задания

1. Доработайте проект, чтобы Птичка управлялась не только клавишей <Пробел>, но и щелчком мыши.

2. Измените скорость движения Труб.

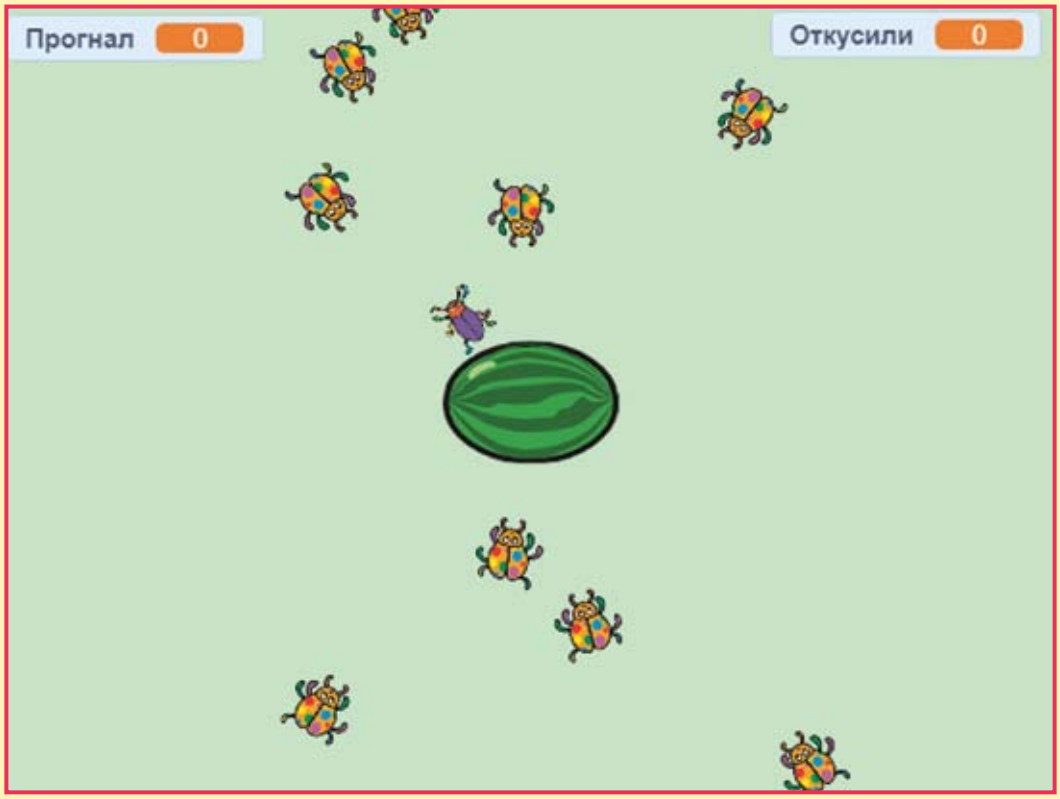
3. Добавьте Монетку, которая при касании Птички будет давать сразу 5 очков.

4. Сделайте так, чтобы при касании Труб игра не прекращалась, а просто очки уменьшались на 1. Условие поражения — количество очков меньше нуля.

5. Добавьте условие победы. Если игрок наберёт 50 очков, то игра заканчивается и появляется надпись «Вы победили!».

ГЛАВА 18. ИГРА «ЗАЩИТА АРБУЗА»

Жук нашёл на грядке Арбуз и ждёт не дождётся, когда же тот созреет. Но про Арбуз узнали мелкие вредители и решили его слопать! Жук должен защитить свою вкусняшку!

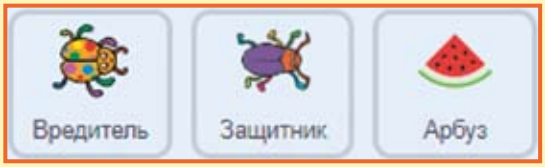


18.1. Создаём спрайты и фон

Создайте новый проект. Удалите Кота.

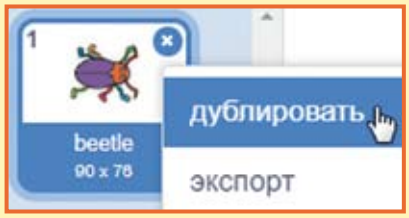
Дело происходит на огороде, поэтому фон должен быть зелёного цвета.

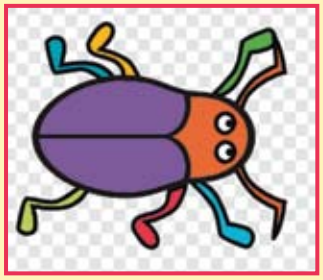
Добавьте в проект спрайты жуков и арбуза и переименуйте их.



Теперь нужно сделать по два костюма каждому жуку и изменить расположение лапок, чтобы анимировать их движение.

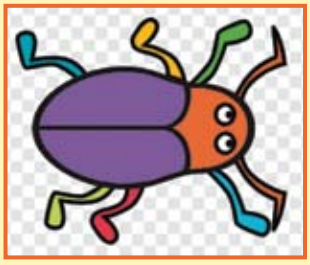
Выберите спрайт Защитника и дублируйте его костюм.





Измените расположение лапок в первом костюме Защитника: слева центральная лапка назад, а передняя и задняя — вперёд, а справа, наоборот, центральная лапка вперёд, а передняя и задняя — назад.

Выделите всего жука инструментом выбора и переместите его направо от центра костюма.



Второй костюм жука-защитника измените похожим образом. Измените расположение лапок: слева центральная лапка вперёд, а передняя и задняя — назад, а справа наоборот, центральная лапка назад, а передняя и задняя — вперёд.

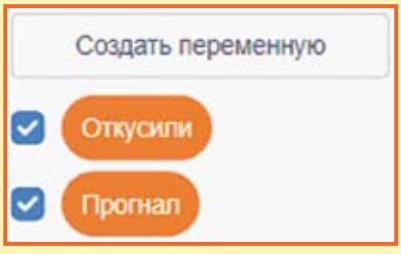
Теперь выделите всего жука инструментом выбора и переместите его направо от центра костюма.

Настало время создать второй костюм спрайту жука-вредителя и изменить расположение лапок.

У вас должны получиться вот такие два костюма.

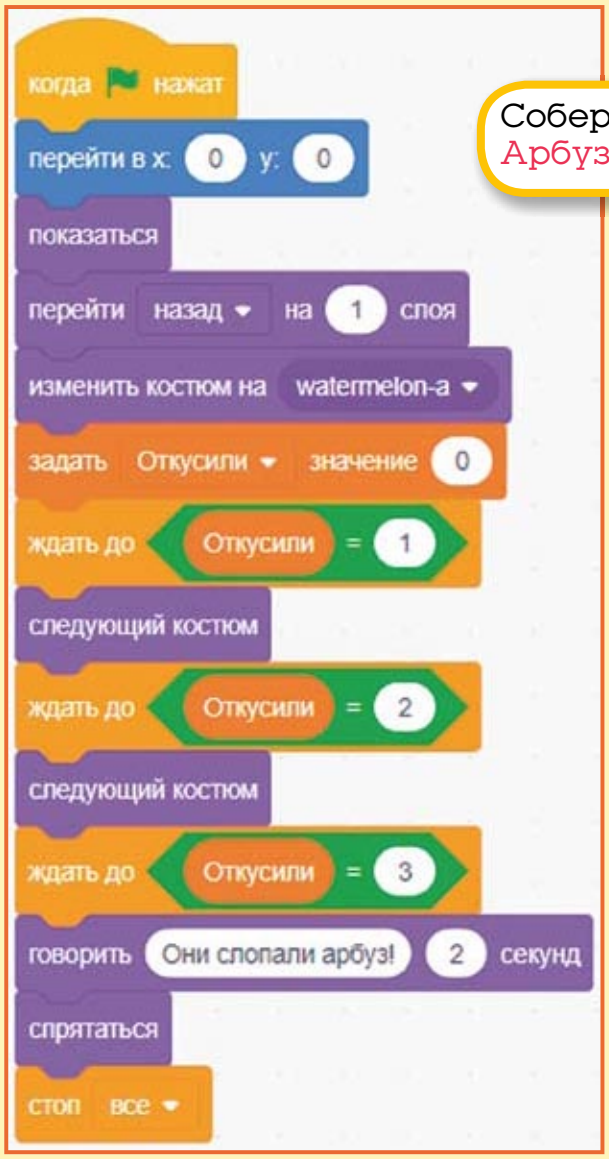
Теперь при движении жуки будут очень мило шевелить лапками, осталось только их запрограммировать.

18.2. Программируем поведение спрайтов



Сначала создайте две переменные.

Переменная **Откусили** будет хранить количество жуков, которые добежали до арбуза и откусили от него кусочек. Переменная **Прогнал** будет хранить количество жуков-вредителей, которых прогнал Защитник.



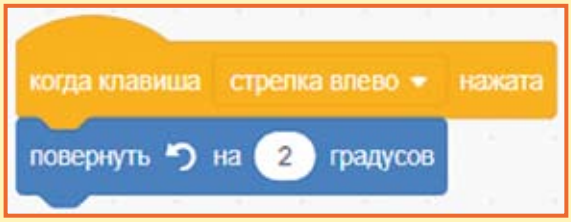
Соберите скрипт Арбуза.

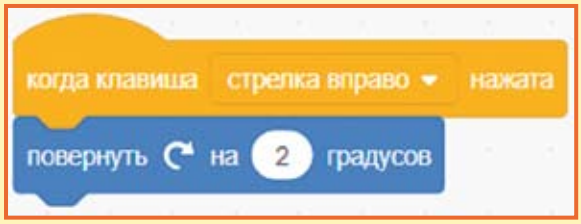
При запуске проекта Арбуз перейдёт в центр сцены, покажется, перейдёт на 1 слой назад и изменит костюм на целый. Затем он задаст начальное значение переменной **Откусили** и будет ждать, пока она не станет равна 1. Когда это произойдёт, Арбуз сменит костюм и будет ждать, пока значение переменной **Откусили** не станет равно 2. Когда это произойдёт, Арбуз снова сменит костюм и будет ждать, пока значение переменной **Откусили** не станет равно 3. Когда это произойдёт, Арбуз скажет, что его слопали, скроется и остановит выполнение программы.

Теперь запрограммируйте Защитника.

При запуске проекта Защитник уменьшится в размере, перейдёт в центр сцены и будет ждать нажатия клавиши <Пробел>. Как только это произойдёт, Защитник грозно помчится вперёд навстречу вредителям, обращая их в бегство. Добежав до края, Защитник снова окажется рядом с Арбузом.

Эти два скрипта управляют поворотом Защитника вокруг Арбуза клавишами-стрелками <←> и <→>.

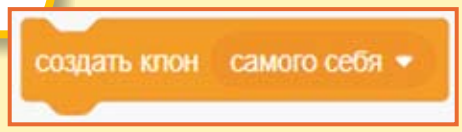




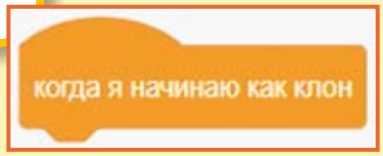
Четвёртый скрипт постоянно проверяет значение переменной **Прогнал**. Если значение равно 10, то вы победили!

Теперь запрограммируйте Вредителя. У этого персонажа всего один спрайт, поэтому для создания армии вредных жуков мы используем клоны. Создадим армию клонов!

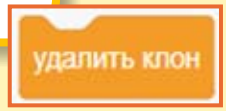
Воспользуемся блоком создать клон.



Для управления клонами служит блок когда я начинаю как клон.



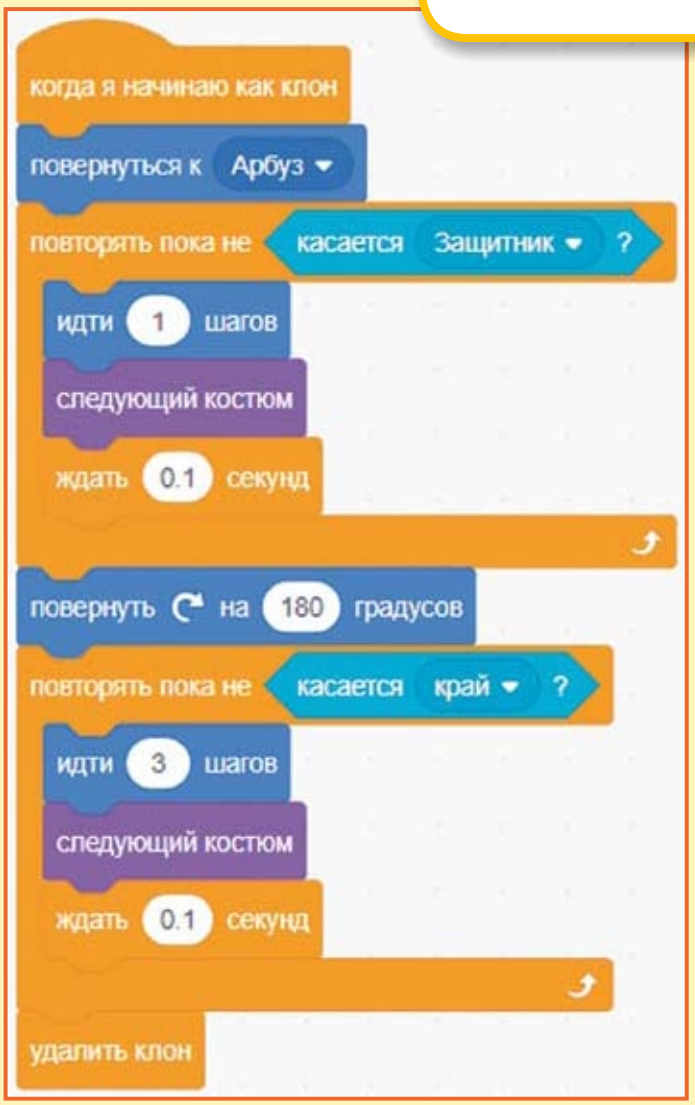
Для удаления клонов предназначен блок удалить клон.



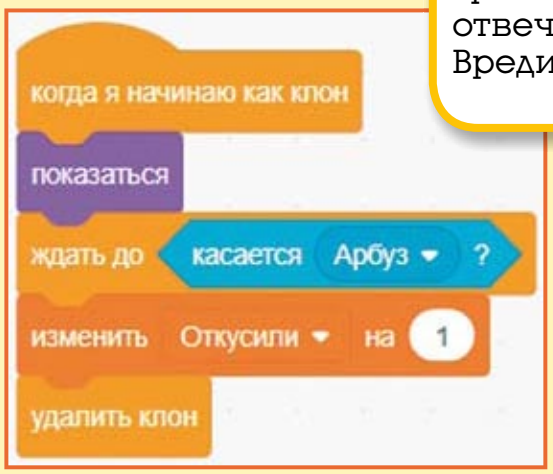
Сделайте
вот такую программу
для Вредителя.

При запуске проекта первый скрипт спрячет Вредителя, уменьшит его размер, а затем создаст 20 пар клонов, с интервалом в 2 секунды. Клоны будут по очереди появляться сверху и снизу, окружая Арбуз со всех сторон.

Второй скрипт отвечает за перемещение Вредителей.



Они всегда поворачиваются к Арбузу и медленно идут, пока не коснутся его. После каждого шага они меняют костюм для создания анимационного эффекта. После того как жук коснётся Защитника, выполнение цикла с условием прекратится, а Вредитель повернётся на 180 градусов и быстро побежит обратно к краю сцены.



Третий скрипт отвечает за касание Вредителями Арбуза.

Как только это произойдёт, значение переменной **Откусили** увеличится на 1, и клон жука будет удалён.

Совет
Никогда не создавайте больше 100 клонов, это очень замедлит выполнение программы.

Протестируйте работу проекта и не забудьте сохранить его.

18.3. Задания

1. Ускорьте движение Вредителей.

2. Сделайте так, чтобы Вредители могли появляться в любом месте сцены.

3. Добавьте на поле кротов, которые будут появляться в случайном месте и отпугивать вредителей.

4. Придумайте своих персонажей и замените костюмы Арбуза, Вредителя и Защитника в соответствии с вашим замыслом (скрипты можно не изменять) — получится совсем другая игра!

ГЛАВА 19. ВИКТОРИНА

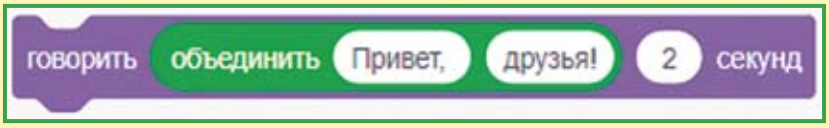
19.1. Работа с текстом

В Scratch можно программировать работу со строками текста, так же как и с числовыми переменными. Строка может быть одной буквой, словом или предложением. Строки можно сравнивать, соединять, произносить и выполнять с ними другие действия.

Для работы с текстом существуют три овальных блока.

Они очень похожи на переменные, их можно вставлять в овальные поля других блоков, например, в блок говорить.

Блок объединить позволяет соединить две строки. Например, если соединить две строки «Привет, » и «друзья!», то получится строка «Привет, друзья!».



Спрайт скажет:

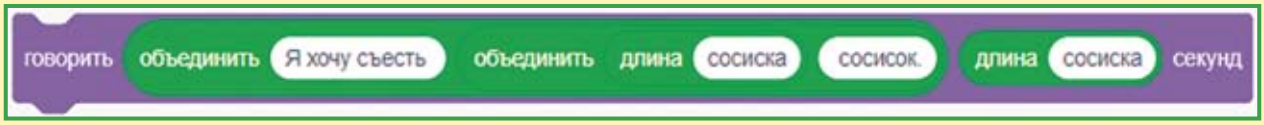
Блок буква создаёт строку, состоящую из одной буквы, вынутой из слова.

Например, если вытащить первую букву из слова «яблоко», то получится строка «я».

Блок длина позволяет определить длину слова.

Например, длина слова «сосиска» равна 7.

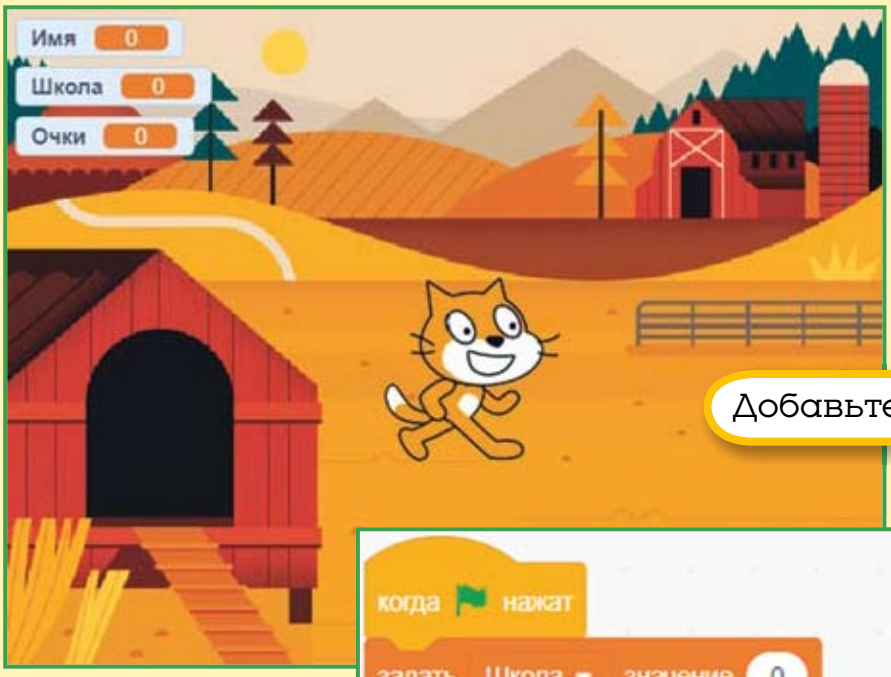
Блоки работы с текстом можно вставлять один в другой, чтобы получить предложения из нескольких слов.



19.2. Простая викторина

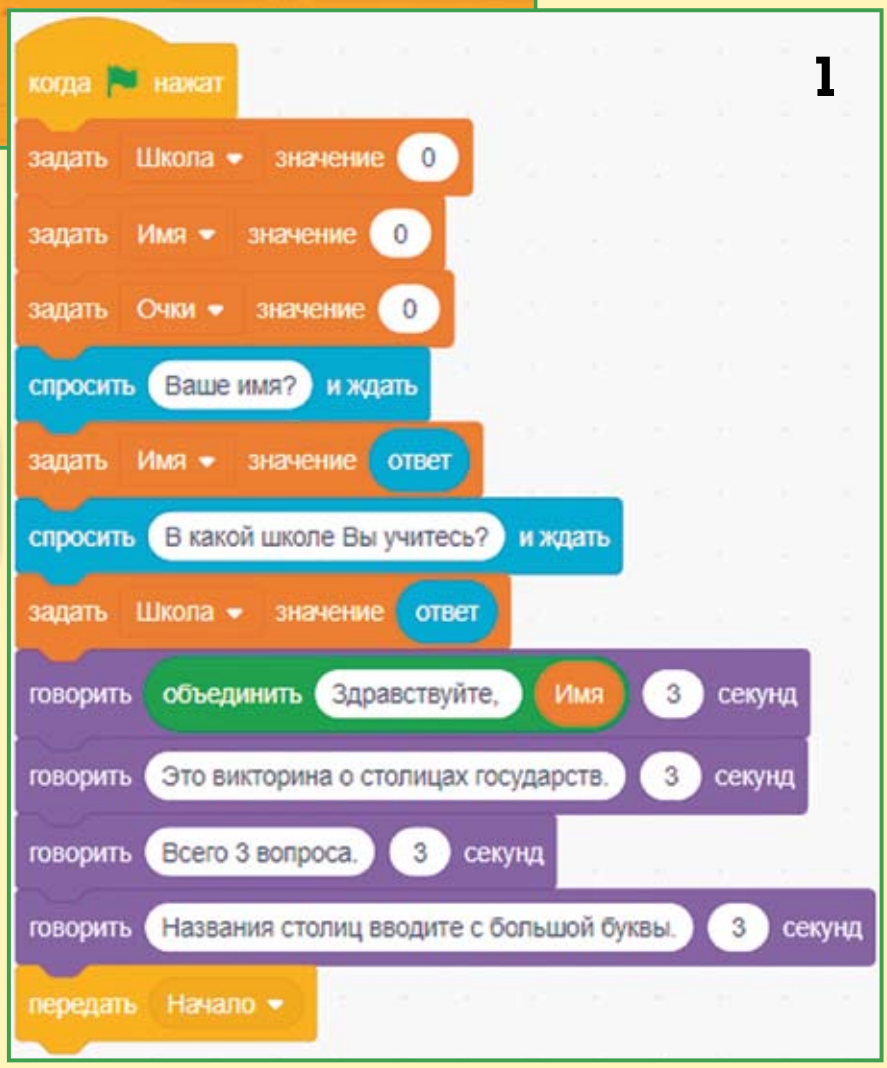
Давайте потренируемся использовать блоки работы с текстом и сделаем игру «Викторина».

Создайте новый проект. Создайте три переменные.

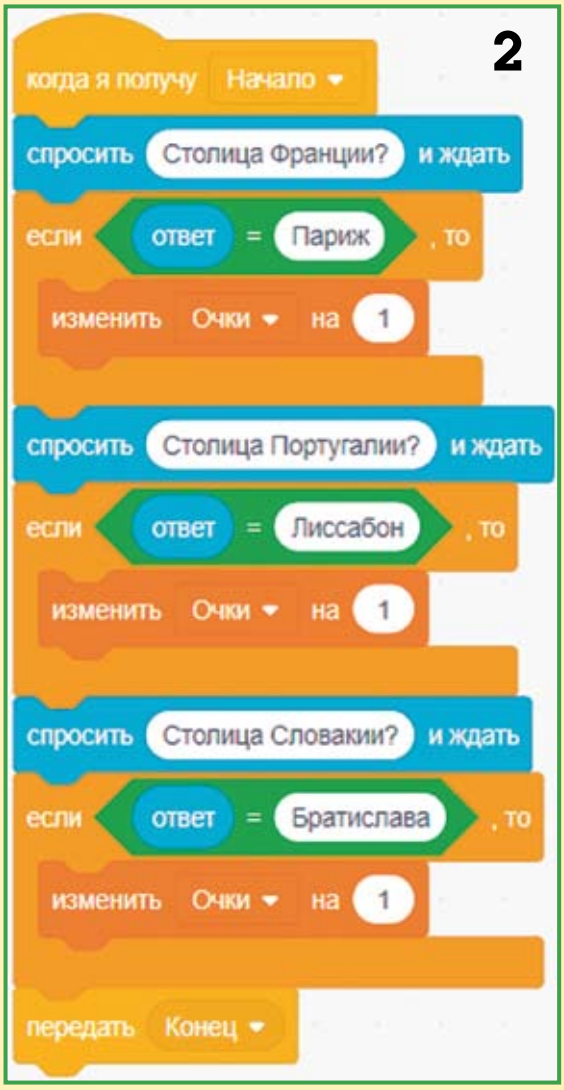


Добавьте на сцену фон.

1



Соберите для Кота вот такие скрипты.

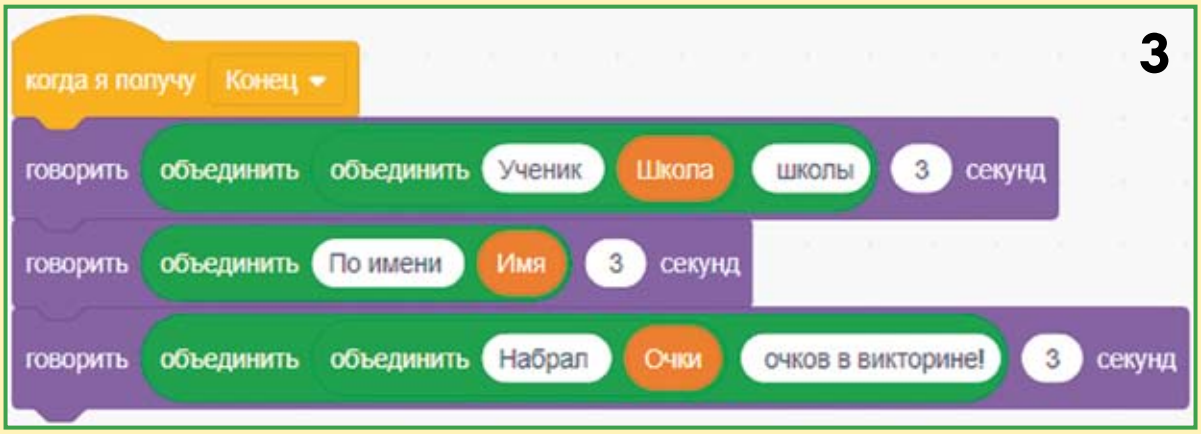


Рассмотрим работу программы. Сначала, как обычно, обнуляются переменные. Далее Кот спросит ваше имя, которое будет помещено на хранение в переменную **Имя**. Затем он спросит, в какой школе вы учитесь, и поместит ответ в переменную **Школа**. Потом Кот произнесёт небольшой монолог и передаст сообщение **Начало**.

Второй скрипт запустится при получении сообщения **Начало**. Кот спросит название столицы Франции. Если содержимое встроенной переменной **ответ** совпадёт со словом «Париж», то значение переменной **Очки** увеличится на единицу. Кот задаст вопрос ещё два раза, а затем передаст сообщение **Конец**.

При получении сообщения **Конец** Кот объявит о результатах викторины. Например, вот так: «Ученик 14 школы По имени Артём Набрал 5 очков в викторине!»

Сохраните проект.



19.3. Задания

1. Доработайте проект, придумайте 10 вопросов.

2. Помимо вопроса о школе, добавьте вопрос о классе, в котором учится игрок.

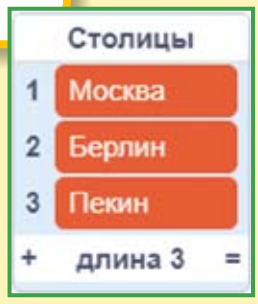
3. Измените проект, задайте вопросы не о столицах, а о чём-нибудь другом — о том, что вы очень хорошо знаете.

4. Добавьте в проект ещё один спрайт, который по результатам викторины поставит оценку. Если все 10 ответов правильные, то «5»; если одна ошибка, то «4»; если две ошибки, то «3»; а если больше двух ошибок, то «2».

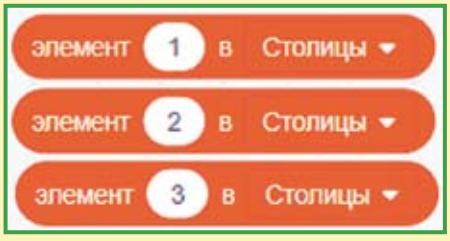
19.4. Викторина со списками

Вы могли заметить, что викторина, состоящая даже из 10 вопросов, представляет собой очень громоздкую программу. А что будет, если вам понадобится задать 50 вопросов? Проект станет очень неповоротливым. Для того чтобы сделать проект короче, можно использовать списки. *Списки* — это наборы пронумерованных переменных.

Например, список с именем **Столицы** хранит 3 значения.



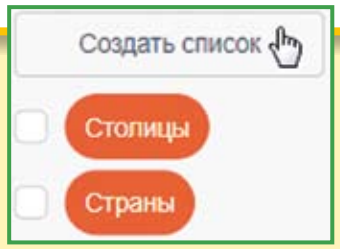
Эти 3 значения, как три переменные с одинаковым именем, но с разными номерами.



Элемент 1 из списка **Столицы** — это «Москва», элемент 2 из списка **Столицы** — это «Берлин», а элемент 3 — это «Пекин».

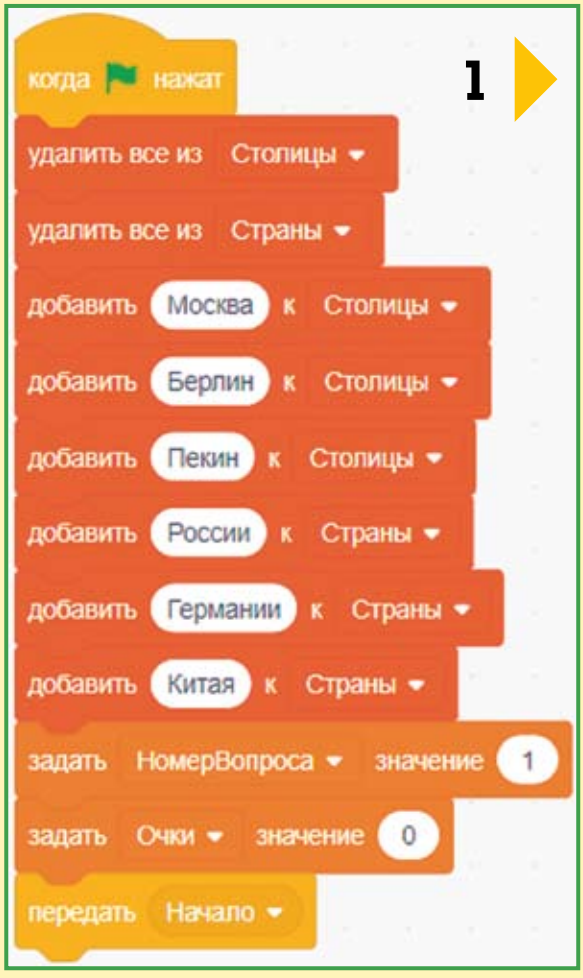
Для того чтобы лучше понять, как работать со списками, создайте знакомую вам викторину о столицах новым способом. Начните с нового проекта.

Сначала создайте список **Столицы**: в блоке **Данные** нажмите кнопку **Создать список** и введите имя списка. Затем создайте список **Страны**.



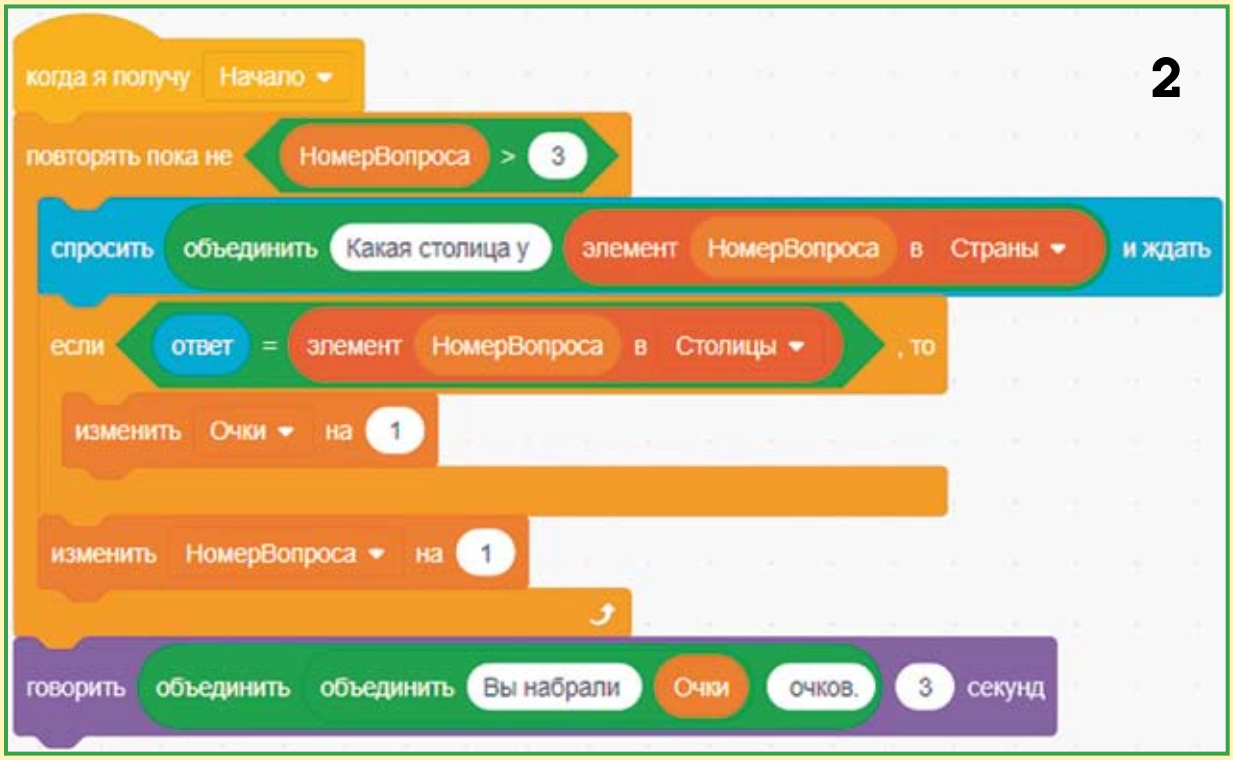
Создайте переменную **НомерВопроса**.

А теперь соберите программу из двух скриптов.



В начале работы программы мы очищаем списки **Столицы** и **Страны** и заполняем их названиями столиц и стран. Затем присваиваем начальные значения переменным и передаём сообщение **Начало**.

Важно!
Порядковый номер страны должен обязательно соответствовать номеру столицы!



При получении сообщения **Начало** переменная **НомерВопроса** будет изменяться в цикле от 1 до последнего вопроса. В цикле будет задаваться вопрос, содержащий название очередной страны из списка, и если ответ совпадёт с названием столицы из соответствующего списка, то переменная **Очки** будет увеличена на 1. После этого **НомерВопроса** увеличится на 1, и при очередном проходе цикла будет задан вопрос о следующей стране. Так будет продолжаться до тех пор, пока **НомерВопроса** не станет больше количества стран в списке. Выполнение цикла завершится, и Кот скажет, сколько очков вы набрали.

Обратите внимание!
Цикл не изменится при увеличении количества стран в списке, изменится лишь число в условии цикла, с которым сравнивается переменная **НомерВопроса**! Только представьте, какого размера был бы скрипт, если бы мы не использовали списки и добавили 100 вопросов!

Сохраните проект и протестируйте его. Предложите друзьям пройти вашу викторину.

19.5. Задания

1. Усовершенствуйте программу: списки не должны показываться на экране.

2. Добавьте ваши вопросы в викторину со списками.

3. Сделайте по два вопроса для каждой страны: про столицу и про континент, на котором находится страна.

СЕКРЕТЫ И СОВЕТЫ

Всё ли вы запомнили? Всеми ли секретами готовы воспользоваться в будущих проектах? Если вдруг что-то забыли — не беда. Вы можете открыть книгу на нужной странице и вспомнить…

ЗАКЛЮЧЕНИЕ

Итак, дорогие друзья, вы познакомились со Scratch и научились программировать простые игры и мультфильмы. Это большое достижение! Однако в этой книге описаны не все возможности программы. Не рассказано о создании собственных блоков, мелодий из нот, о работе с видео и о многих математических операторах. Но, как говорится, дорогу осилит идущий. Постарайтесь разобраться с этими темами самостоятельно, используя библиотеку знаний о Scratch (Scratch Wiki), расположенную по адресу https://wiki.scratch.mit.edu. Вопросы можете задавать на русскоязычном форуме о программировании на Scratch, расположенном по адресу https://scratch.mit.edu/discuss/27/. Также можете задавать вопросы автору по электронной почте scratch.book@ya.ru.

Удачи! И до новых встреч!

СОДЕРЖАНИЕ