# Методические рекомендации к проведению урока

Блок Данные. Создание переменных

## Основные понятия, рассматриваемые на уроке

* Переменные
* Списки
* Имя переменной(списка)
* Значение переменной (элемента списка)
* Индекс элемента списка

## Используемое на уроке оборудование

* интерактивная доска
* проектор
* компьютер учителя
* компьютеры учеников
* программа ScratchDuino2 / Robboscratch2

## Используемые на уроке электронные образовательные ресурсы

* не используются

## Особенности изложения содержания темы урока

Повторение

Здравствуйте, ребята, приветствую вас на нашем очередном занятии.

Давайте вспомним, что мы изучали на прошлом занятии? Педагог слушает ответы детей, задает наводящие вопросы и подводит итоги обсуждения. Для того, чтобы создать игру “Угадай число” нужно научить компьютер загадывать числа, если у нас команда, которая поможет это сделать? (Дети называют генератор случайных чисел). Учитель демонстрирует три варианта и просит указать верный:







Ответ: все варианты - правильные.

## Переменная

От понятия числа переходим к понятию “переменная”. Что называют переменной? В математике при решении уравнений вы использовали переменные, величины, значение, которых нужно найти. В программировании, переменная - это именованная (имеющая название) область памяти компьютера для хранения какой-либо информации.

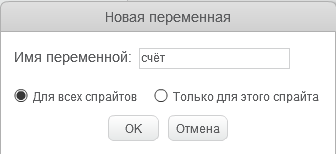
Нарисуем на доске три банки. Желтая - для конфет, синяя - для печенья, красная - для сухофруктов. А что будет, если, например, в желтую банку положить печенье? Всё перемешается, и будет путаница.



Вот банки - это переменные. Если на банках написать названия, это будут названия переменных. А содержимое банок - сухофрукты, конфеты, печенья - значения переменных.

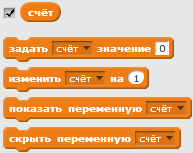
В Robboscratch мы можем создать переменные. Для этого зайдем в блок Данные и нажмем кнопку .

Назовем переменную счёт (счётчик). Переменная будет считать количество раз, которое кот коснется краёв сцены в программе.



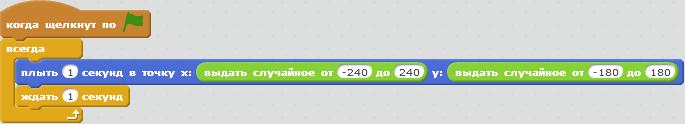
Переменная может быть как для всех спрайтов (так называемая глобальная переменная), так и для какого-то одного (локальная переменная). Оставим по умолчанию переменную “счёт” глобальной.

Если переменная создана, то в блоке Данные появляются новые команды. Пока достаточно первых двух: задать какое-то начальное значение (как говорят программисты, инициализировать переменную) и изменить это значение на сколько-то.



Если стоит галочка в поле команд у переменной, то она отображается на сцене (иногда это удобно, например, в играх при счёте очков). Уберем галочку. Что произошло с переменной? Теперь её не видно на сцене. Опять поставьте галочку и верните переменную на место. Переместите переменную в другую часть сцены с помощью левой клавиши мыши. Перемещается ли она?

Создайте следующий скрипт для кота.

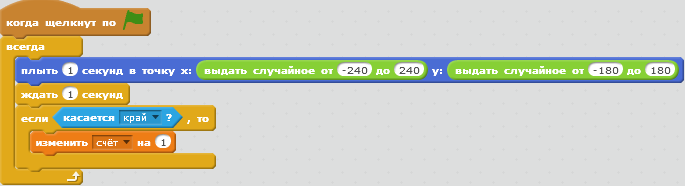


*Предложите учащимся прокомментировать код (повторение материала предыдущего урока).*

Сделаем так, чтобы счётчик (переменная счёт) для этого кота считал, сколько раз тот коснётся краёв сцены.

Нам понадобится сенсор и это делается следующим образом. В существующий скрипт вставляется условие: если касается края, увеличиваем переменную на 1.

Получается следующий скрипт:



Увеличивать переменную можно на любое другое число, а также переменную можно уменьшать, написав отрицательное число в окошко. Вспомните, мы это делали в программе Прыгающий мяч, на втором занятии.

Выполните программу несколько раз и понаблюдайте за счётчиком-переменной (для этого мы и оставили её на сцене). Что происходит с переменной при каждом новом запуске программы? Ничего. А нам совершенно не нужно, чтобы переменная сохраняла своё значение при каждом новом запуске, нам бы хотелось, чтобы счётчик обнулялся.

Задаем вопрос учащимся: как это можно сделать? (Выслушиваем их предложения). Правильно, перед бесконечным циклом необходимо вставить команду **Задать счёт значение 0** из блока Данные.

Сделайте это и запустите программу несколько раз. Теперь переменная обнуляется.

Предложите разным учащимся самостоятельно измените программу так, чтобы переменная увеличивалась на разное значение при каждом касании экрана. Убедитесь, что у всех это получилось.

На следующем шаге, предложите учащимся заменить программу так, чтобы переменная увеличивалась на 1 при каждом касании мыши.

*Задание для быстро успевающих:* Написать программу (или изменить существующую) таким образом, чтобы при касании мяча (добавить соответствующий спрайт) котом (который также пока движется в произвольном порядке) переменная увеличивалась на 1.

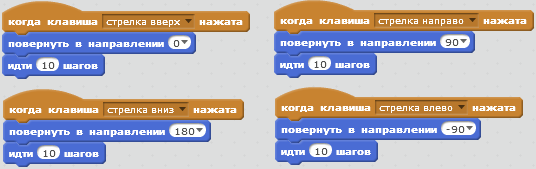
Теперь добавим несколько предметов-спрайтов на сцену.

Например, так:



Коту нужно сделать управление с кнопок *(попросите учащихся вспомнить, как это делается, повторение первых уроков).*

*Также напомните про кнопку  (дублировать), позволяющую ускорить процесс создания однотипных частей скрипта и однотипных спрайтов(костюм можно изменить в любое время).*

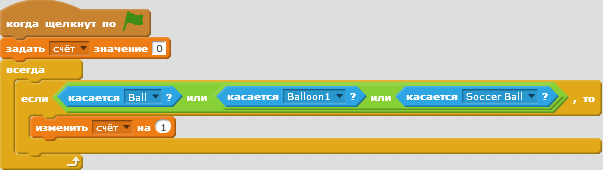


Проверьте правильность выполнения программы. Кот должен управляться полностью с помощью 4 кнопок.

Далее, измените программу: когда кот касается одного из предметов, переменная увеличивается на 1. Расставьте спрайты по сцене подальше друг от друга и немного уменьшите так, чтобы они были соразмерны коту.

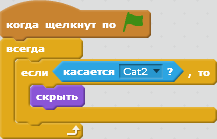


*Далее вместе с учащимися напишите часть скрипта для кота, благодаря которой переменная будет увеличиваться при касании предметов котом.*



***Важно!!!*** *Ещё раз проговорите, для чего нужно обнулять переменную в начале каждой программы. А также напомните значение логических операторов ( команда* ***… или …*** *).*

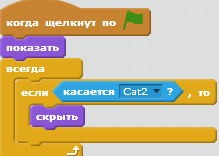
Выполните программу. Всё работает, но что происходит с переменной? Как только кот касается одного из предметов, переменная начинает очень быстро увеличиваться, потому что происходит постоянное касание. Изменить это можно, например, следующим образом. Для спрайтов-предметов напишем скрипт удаления (исчезания) при касании их котом.



Самостоятельно напишите скрипты для оставшихся спрайтов *(будет неплохо, если учащиеся вспомнят, что можно скопировать код другому спрайту).* В итоге последний приведённый скрипт должен быть одинаков для всех спрайтов-предметов.

Проверьте, как работает программа. Все спрайты должны удаляться при касании котом.

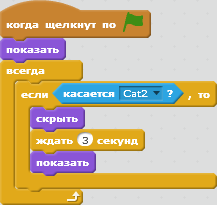
Спрайты не появляются при каждом новом запуске программы. Самостоятельно исправьте эту проблему. Скрипты для всех спрайтов-предметов должны быть такими:



Теперь спрайты должны появляться при каждом запуске программы.

Нас не устраивает, что спрайты в одной игре исчезают и больше не появляются. Это ведь значит, что в одной игре можно набрать только 3 очка. А нам бы хотелось большего количества.

Значит, нужно сделать так, чтобы через какое-то время после исчезновения спрайты появлялись вновь. Следующим образом:



Осталось сделать 2 вещи:

Первое - сделать так, чтобы спрайт показывался не ровно через 3 секунды, а через произвольное время (2, 3, 4 или 5 секунд).

Второе - начинать программу кот должен в такой точке, чтобы он не касался предметов и изначально спрайты не удалялись - иначе будет неинтересно.

Предложите учащимся самостоятельно сделать игру более интересной и динамичной.

***Дополнительно*** *для быстро успевающих:*

*\** добавьте спрайты,

\*\* звуки,

\*\*\* различные условия.

## 2. Создание списков

Рассмотрим ещё одну возможность в блоке Данные - создание списков (кнопка Создать список).

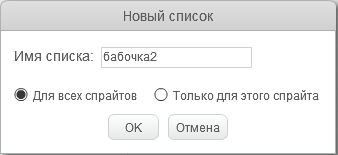
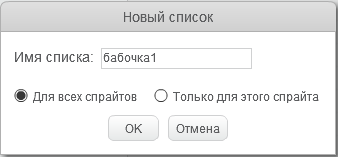
Создадим новый файл и добавим два спрайта: Butterfly2 и Butterfly3.

Создадим диалог между бабочками. Вспомните, что вы использовали на одном из последних уроков для написания диалогов (передачу сообщений). Помните, что код получился очень длинным? С помощью списков можно уменьшить длину кода.

Разверните правую бабочку так, чтобы бабочки смотрели друг на друга.

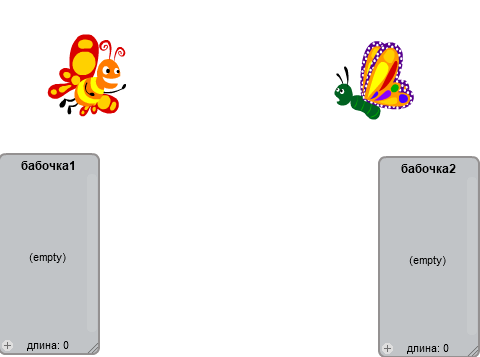
Список - это перечисление чего-то. Например, список детей в классе.

Создадим список бабочка1 (нажав кнопку Создать список). Самостоятельно создайте список бабочка2.



Первый список будет отвечать за фразы левой бабочки, второй - за фразы правой.

На сцене появились прямоугольные поля, имеющие название (можно менять их размер и передвигать как угодно по полю). Также появились новые команды в поле списков.



В середине прямоугольного поля написано empty. Это значит, что списки пустые (в них нет элементов). Также об этом говорят нижние строчки (длина: 0).

Длиной списка называется количество элементов в списке.

Например, сколько элементов в списке *(написать на доске)*

Маша

Даша

Петя

Вова

Витя?

Пять - это и есть длина списка, а порядковый номер элемента его индекс.

Выполните задание 3 в РТ.

Изучим один из способов добавления элементов в список - вручную. Для этого в левом нижнем углу каждого прямоугольника есть плюсик, который нужно нажать.

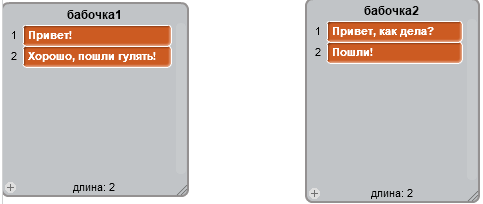


Появится строчка, в которую можно написать элемент списка. Напишите “Привет!”.

Второй бабочке напишите первый элемент “Привет, как дела?”.

Допишите списки. Каждый элемент - это очередная фраза спрайта.

Например, так:



Обратите внимание, что значение длины списков изменилось - стало равно количеству фраз каждого спрайта.

Если какой-то элемент списка нужно удалить, нужно нажать на него и нажать на крестик, появившийся справа.

Скрыть список можно точно также, как и переменную.

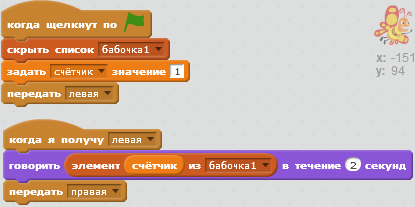
Для первого спрайта:



Для второго аналогично. Напишите код самостоятельно.

Нам понадобится переменная. Назовем её счётчик. Она будет отвечать за номер фразы в списке(индекс).

*Напишите следующие коды. Попросите учащихся их также написать и прокомментировать, знаний у них уже достаточно.*

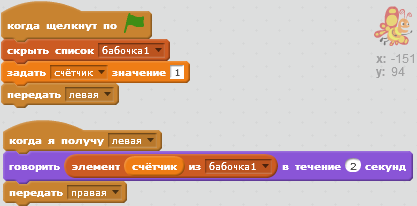




Нажмите на зеленый флажок и посмотрите, как программа работает.

Что работает неправильно? Спрайты обмениваются первыми сообщениями и следующие сообщения не появляются. Почему? Подсказка: для чего вводили переменную (за что она отвечает)? И что с ней нужно сделать?

Код программы должен получиться таким:





Проверьте, как работает программа. Понаблюдайте за изменением значения переменной.

Сначала всё изменяется правильно, но когда диалог заканчивается, переменная продолжает увеличиваться. Как-то нужно это исправить.

Для этого рассмотрим операторы сравнения (блок Операторы):



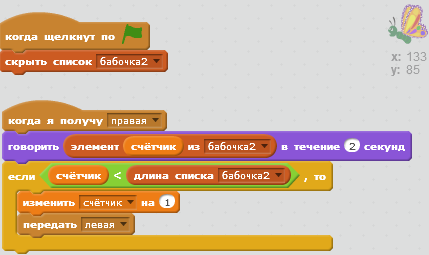
Если известно, что a<b, значит, a находится левее b на числовой прямой (нарисовать на доске числовую прямую). Например, 3<7, 5<10 и т.д.

Вместо чисел (или одного из них) могут стоять и другие объекты, например, переменные.

Аналогично другие два оператора, вместо чисел можно поставить другие объекты.

Для того, чтобы устранить проблему в программе, добавим проверку условия: если значение переменной меньше длины списка, то только в этом случае увеличиваем значение переменной и передаём сообщение.

Итоговый скрипт для правой бабочки:



***Дополнительно*** *для быстро успевающих:*

*\** добавьте смену фона в середине диалога бабочек (если значение счетчика = 3, то фон меняется)

\*\* аналогично предыдущему пункту добавьте звуки, при определенном условии (например, в начале диалога начинает звучать музыка при значение счетчика = 1)

\*\*\* самостоятельно добавьте, по своему усмотрению, другие условия, которые позволят оживить и сделать более содержательным и интересным диалог бабочек и данный скрипт. Можно организовать диалог и для других персонажей, сменив костюмы или добавив спрайтов на сцену.

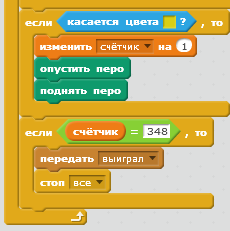
Финальная игра

Откройте файл с итоговой игрой. *Учитель также открывает свою игру.*

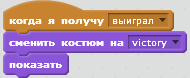
Создадим условие выигрыша. Игрок выиграет, если коснётся всего корма, конкретно - 348 жёлтых кружочков.

Создадим переменную счётчик. И напишем условие: если переменная равна 348, то игрок выиграл и для спрайта надпись необходимо создать соответствующий скрипт.

Добавленные команды для Пакмена (не забудьте в начале обнулить переменную!):



Добавленные команды для спрайта Надпись:



Файл сохраните.

*Напомните учащимся о том, что в рабочей тетради нужно зафиксировать изменения, сделанные в своей игре.*

*В конце урока попросите учащихся раскрасить соответствующего робота.*