**КАК СОЗДАТЬ ШУТЕР С ВИДОМ СВЕРХУ-ВНИЗ С НУЛЯ?**

*Мануал рассчитан на людей, имеющих представление о базовых правилах программирования.*

## ПРЕДИСЛОВИЕ

Мечта создать свою игру преследует почти каждого геймера, и каждый знает, что это трудно. Первая установка Unity, Godot и прочих - тест на терпение. Первые уроки программирования - легкотня, применение знаний на практике - снова сложно. Создание игр это трудно, в десять раз комплексней уроков информатики. Вы не станете популярным разработчиком, ваша игра не стрельнет, если вы поймете, что здесь написано. Это не демотивация, это лишь предупреждение для энтузиастов.

Сейчас я пользуюсь шансом рассказать о том, как я пришла к разработке. С самого детства я была окружена популярными тайтлами, GTA: San Andreas была установлена на моем планшете, все диски старшего брата с играми были рано или поздно вставлены в дисковод ноутбука на каникулах у бабушки, а игры скачаны. Первая консоль, ретро комбайн в коробке, эмулятор NES, SNES, Sega Megadrive, открыла мне мир классики видеоигр. Домашний компьютер и детские Lego игры затянули меня не меньше, чем летсплееры в юном возрасте. Все мое детство, юность и подростковый период игры были основным источником влияния. Моей мечтой было сделать что-то похожее, и, наконец, я пришла к тому, что могу это сделать. В этом коротком мануале, я хочу поделиться основами основ с теми, кто не знает с чего начать. Знания, полученные здесь, могут быть использованы не только в TDS проектах.  
 Информация, которую я предоставляют может быть использована для создания таких жанров как платформер, визуальная новелла и прочих.

# НАЧАЛО

Вам понадобиться спрайт персонажа, которым игрок будет управлять. Или же несколько, если вы хотите, чтобы спрайты менялись в зависимости от направления персонажа. Для этого в папке Sprites создаем спрайт нажатием ПКМ.

Этот код пишем в событии Step (которое происходит постоянно)  
  
if keyboard\_check\_direct(ord("W")){

y-=5

}

if keyboard\_check\_direct(ord("S")){

y+=5

sprite\_index = Sprite12

}

if keyboard\_check\_direct(ord("D")){

x+=5

sprite\_index = Sprite10

}

if keyboard\_check\_direct(ord("A")){

x-=5

sprite\_index = Sprite11

}

Основной спрайт выбирается в специальном окне, если он не выбран, то будет текст “No Sprite”.

В Game Maker ось X остается привычной, правее +, левее -. Однако ось Y инвертирована, то есть выше -, а ниже +.

В коде написано, что во время того как мы удерживаем одну из кнопок меняется положение по одной из осей и спрайт (опционально).

Далее, вы вероятно захотите сделать мир для нашего игрока. Для этого скачиваем (или рисуем) тайлсет, то есть набор текстур на одном изображении. Для этого в папке Tileset по схожему принципу со спрайтами создаем тайлсет, выбираем название и выбираем ранее загруженный спрайт тайлсета (то есть сначала загружаем спрайт, а потом уже выбирает его как тайлсет). Настройка тайлсетов проста, нужно выбрать ширину (tile width), высоту (tile height) и, если такие имеются, пропуски(spacing). Далее, создаем комнату в папке Rooms (либо уже заходим в стандартную комнату, уже созданную программой.) Перемещаем мышкой тайлсет, вам предложат создать специальный слой для ТС, соглашаемся.

***Для каждого типа файлов должен существовать слой (их может быть несколько). Для объектов это Instances, спрайтов Assets, тайлсетов Tileset.***

Далее выбираем кнопкой мыши нужную часть тайлсета (квадратик) и рисуем. Если требуется создать новый слой, нажимаем на плюс и выбираем тип (основные представлены выше).   
  
Чтобы наш игрок не мог ходить по домам, создаем создаем слой Instances и объект, обязательно нужно закрасить объект каким нибудь цветом, иначе его свойства не будут работать. Теперь у игрока создаем событие Collision и выбираем объект, через который игрок не сможет пройти.  
  
Пишем:

x = xprevious

y = yprevious

Это означает что всякий раз, когда игрок сталкивается с объектом, его X и Y будут равны X и Y в первый момент столкновения (координаты игрока попросту не будут меняться). Теперь располагаем и при необходимости растягиваем объект по всей “непроходимой площади”, а в списке всех слоев нажимаем на глаз, чтобы сделать наш слой невидимым.  
  
Теперь приступим к спавнеру врагов и регулировке игровых правил, создаем 3 объекта с названием Spawner 1, 2, 3, а также главный контроллер, который будет отвечать за игровые правила, так как будет проще разместить их всех в одном месте, если дальше потребуется их корректировка.

Пока что займемся спавнером, для этого создаем событие Create - в нем создаются и хранятся переменные, которые можно разделить на локальные и глобальные. Первые доступны только в объекте, в котором они созданы. Глобальные можно использовать везде.  
  
Событие Create:  
spawn = 1

timer = 300

global.spawnnum = 0

Событие Step:

if spawn = 1{

global.spawnnum = irandom\_range(1,3)

spawn = 0

alarm[0]= timer

}

Событие Alarm[0] (событие которое происходит через указанное время, которое измеряется в FPS):

spawn = 1

switch global.spawnnum{

case 1:

global.spawnnum = 0

instance\_create\_depth(SPAWN1.x, SPAWN1.y,0, p2)

break;

case 2:

global.spawnnum = 0

instance\_create\_depth(SPAWN2.x, SPAWN2.y,0, p2)

break;

case 3:

global.spawnnum = 0

instance\_create\_depth(SPAWN3.x, SPAWN3.y,0, p2)

break;

}

О том, что здесь происходит: переменная spawn отвечает за активацию спавнера. Мы отключаем ее и по таймеру она включается в событии alarm, чтобы враги не появлялись бесконечно - это неиграбельно и нагружает игровое устройство.

global.spawnnum - номер спавнера (1 из 3 в моем случае), irandom\_range позволяет получать случайно число из указанного списка, первое число - минимальное значение, второе - максимальное включительно.

timer - время, отсчитываемое до выполнение действий из события alarm.

instance\_create\_depth - создание объекта, первое - координата X, второе - координата Y, третье - глубина (depth, то есть слой, посмотреть значение можно в редакторе комнат, для этого нужно нажать на слой), четвертое - нужный объект. Там, где мы пишем координату, можно выбрать координату какого либо объекта, координату в целом (числами), или же переменную. С координатами можно проводить простейшие математические выражения - наиболее ходовые сложение и вычитание, их мы рассмотрим чуть позже.

***Не забудьте расположить все объекты, включая Main, в комнату, иначе код не будет действовать.***

Дальше, сделаем так, чтобы враг двигался к игроку, не мог ходить по зданиям.

Событие Create в Main:

global.spd = 0.75

Событие Step в объекте врага:

move\_towards\_point(p1.x,p1.y, global.spd)

Событие Collision в объекте врага:

x = xprevious + 10

y = yprevious + 10

О том, что здесь происходит: в переменной global.spd мы задаем скорость врага.  
move\_towards\_point - заставляет объект двигаться к какой либо координате. Первое значение - координата X, второе - координата Y, третье - скорость (не обязательно создавать отдельную переменную).

p1 - название объекта игрока.

Теперь добавим главное в любом шутере - пушку и возможность стрелять.

Создаем два объекта - пуля и пистолет.

Событие Step в объекте пистолета:

x = p1.x+3

y = p1.y+1

switch global.gunposition{

case 0:

sprite\_index = revolver

break;

case 1:

sprite\_index = revolver\_1 //спрайт, в котором пистолет направлен в другую сторону

x = p1.x-5

break;

case 3:

sprite\_index = revolver

y = p1.y-1

break;

}

Событие Create в Main:

global.gunposition = 0

Также редактируем событие Step в объекте игрока:  
if keyboard\_check\_direct(ord("W")){

y-=5

**global.gunposition = 3**

}

if keyboard\_check\_direct(ord("S")){

y+=5

sprite\_index = Sprite12

**global.gunposition = 0**

}

if keyboard\_check\_direct(ord("D")){

x+=5

sprite\_index = Sprite10

**global.gunposition = 0**

}

if keyboard\_check\_direct(ord("A")){

x-=5

sprite\_index = Sprite11

**global.gunposition = 1**

}

Координаты пистолета подобраны под спрайты моего игрока, чтобы он не выглядел летающим, у вас они могут (скорее всего будут) отличаться.

Событие Step в объекте пули:

move\_towards\_point(mouse\_x,mouse\_y,10)

alarm[0] = 10

Событие Alarm в объекте пули:

instance\_destroy(self)

Событие Collision в объекте пули:

instance\_destroy(self)

Mouse\_x и mouse\_y - автоматически установленные значения, которые равны координатам мыши.  
Instance\_destroy() уничтожает объект, написанный в значении, значение self приводит к тому, что уничтожается тот объект, в котором это прописано.

Чтобы избежать бесконечного преследование пулей курсора, при столкновении с курсором, пуля также уничтожается.

Теперь сделаем возможным стрельбу.

Событие Step в объекте игрока:

if keyboard\_check\_pressed(ord("E")){

instance\_create\_depth(p1.x+3, p1.y+1,0,gun)

global.gun = 1

}

if keyboard\_check\_pressed(ord("Q")){

instance\_destroy(gun)

global.gun = 0

}

Это позволит “достать” и “убрать” оружие по нажатию кнопки. Переменная global.gun такой же переключатель. как и у спавнеров, значение 1 включает возможность стрелять, 0 выключает.

Событие Step в объекте Main:

if global.gun = 1 and mouse\_check\_button\_pressed(mb\_left){

instance\_create\_depth(gun.x,gun.y,100,bullet)

}

Create в объекте Main:

global.gun = 0

global.kills = 0

Событие Collision с пулей в объекте врага:  
instance\_destroy(self)

global.kills+=1  
  
Добавим статистику, которая будет с ростом убийств ускорять процесс спавна врагов. Для того, чтобы игрок мог следить за количеством убийств, нужно чтобы он их видел.  
  
Событие Draw в объекте Main:

draw\_text(p1.x,p1.y-20, "Kills: "+string(global.kills))  
  
string() преобразует числовое значение в строковое. Первое значение - координата X, второе - координата Y, третье - текст.

Событие Step в объекте Main:

switch global.kills{

case 5:

timer = 200

break;

case 15:

timer = 100

break;

case 30:

timer = 50

break;

}

Уменьшается значение таймера при достижении определенного количества убийств.

Также можно заменить спрайт курсора на любой другой. Для этого нужно создать спрайт желаемого курсора и объект.

Событие Step в объекте мыши:

x = mouse\_x

y = mouse\_y

Событие Step в объекте Main:

cursor\_sprite = mouse