ВИДЕОИГРЫ
ГЛУБОКОЕ ПОГРУЖЕНИЕ

СЛАВА ГРИС

СДЕЛАЙ ВИДЕОИГРУ ОДИН

И НЕ СВИХНИСЬ!

Слава Грис
Сделай видеоигру один и не свихнись

…Он был полон великих идей, но не работал в своей жизни ни дня.

Divinity: Original Sin,

надпись на надгробном камне

В оформлении издания использованы работы автора

© Слава Грис, 2023
© ООО «Издательство АСТ», 2023

https://t.me/it_boooks

0. Экран загрузки

Я знал множество людей, создающих видеоигры без поддержки команды. Они посещали сходки разработчиков видеоигр; кое-кто из них появлялся на конференциях, а иногда их можно было встретить на необъятных просторах Сети. Их очень много. Я и сам – разработчик-одиночка.

Я нередко участвовал в пылких спорах о том, возможно ли создать в одиночку нечто столь комплексное, как видеоигра? Она же состоит из графики, музыки, сценария, кода, анимаций, игрового дизайна! Способен ли всего один человек справиться с таким неизмеримым количеством работы?

Многие приходили в своих рассуждениях к положительному ответу: существуют примеры, когда старания соло-разработчиков порождали чудесные и целостные игровые миры. Stardew Valley, Undertale, Cave Story – за каждым из этих проектов стоит всего по одному человеку. Их пример вселяет в людей твердую уверенность, что сделать хорошую игру в одиночку – возможно. И, оглядываясь на эти примеры, люди воодушевленно берутся воплощать свои идеи в реальность, параллельно обучаясь всему и сразу.

Но спустя время знакомые мне разработчики-одиночки начинали пропускать мероприятия, конференции и сходки. Их социальные сети, где они гордо делились своими успехами, покрывались электронной пылью. И вот уже последней записи на странице их проекта исполняется год, а сами авторы удаляют меня из друзей, потому что я перестал входить в их круг общения. Некогда мечтательных и уверенных в себе разработчиков игр уносили с собой суета будних дней и совсем уже другая работа. Мечта выпустить идеальную игру так и осталась мечтой. Друзья-разработчики больше им не нужны.

Сформулировать программу действий, которая позволит вам освоить столь необычное ремесло, как создание игр в одиночку, не очень-то и сложно: мы займемся чем-то подобным на страницах этой книги. Но, наметив себе путь, начинающие разработчики неистово хватаются за рисование, написание кода, создание музыки, а спустя какое-то время,

подобно старой лампочке, мигают пару раз и больше не светят в сторону видеоигр.

Ключевая ошибка всех, кто бросил свой проект, едва начав, заключается в том, что в самом начале пути был поставлен неправильный вопрос и велась работа лишь на одном фронте. Утвердительный ответ на «возможно ли разработать игру в одиночку?» побуждает новичков погружаться в техническую часть создания видеоигр, в то время как моральная и психическая часть остается нетронутой.

Правильный вопрос перед вступлением на красочный путь разработчика звучит так: «возможно ли создать игру в одиночку и не свихнуться?» Чтобы довести проект до ума, нам недостаточно накопить технических знаний и художественных навыков.

Нам нужно оставаться мотивированными. Нам нужно избегать эмоционального выгорания. Нам нужно запастись терпением, выдержкой и знать все о самодисциплине и силе воли. Работа по внезапному наитию едва ли будет доведена до конца, а если и будет, то этот процесс займет у вас немыслимо длительный период.

Инструмент для разработки видеоигр – это не только компьютер, но и наши нервы, голова, психика, установки и наше окружение. Именно они помогут не пасть духом; именно в них скрыты наши силы; именно там мы найдем способ не сойти с долгого и увлекательного пути.

Пути к игре своей мечты.

https://t.me/it_boooks

1. Раньше было лучше (?)

Когда в компаниях, где меня никто не знает, я говорю, что зарабатываю на жизнь разработкой игр в одиночку, я обязательно сталкиваюсь с убеждением, что моя деятельность почти фантастична, а сам я или лжец, или пришелец с другой планеты. Уверенность в том, что для разработки современной игры нужна огромная команда, слишком уж плотно укоренилась в массовом сознании. Студии по разработке видеоигр нанимают бессчетное количество людей. Титры в играх становятся все длиннее. Команды разработчиков перестают помещаться на одном фотоснимке. В игровой индустрии теперь очень много народа, и это почти никого не удивляет.

А меня – удивляет. И вас, надеюсь, это тоже удивит, когда мы познакомимся с некоторыми аргументами в пользу того, что «один в поле – вполне себе разработчик», и нечего удивляться, когда в компании кто-то говорит, что делает игры один. Я не занимаюсь волшебством.

За время существования такого искусства, как «создание видеоигр», было разработано множество великолепных проектов. И если о культовом статусе некоторых современных произведений множество людей пылко ругаются на форумах, то почетный статус таких вещей, как Mario и Final Fantasy, – неоспорим.

Давайте на короткое время вернемся в конец восьмидесятых, чтобы после такого путешествия свежим взглядом окинуть современную игровую индустрию и подумать о нашем месте в ней.

На заре становления индустрии видеоигр все платформеры или разворачивались на одном экране, или же вынуждали персонажа двигаться вверх. Но тут свершается революция – рождается такая легенда, как **Super Mario BROS**. Миллионы игроков пускаются в оригинальное приключение «слева направо». Революционность Марио на NES (в России продавали клон этой приставки под названием Dendy) заключалась не только в направлении движения, но и во многих оригинальных дизайнерских решениях, которые требуют куда более детального анализа, чем можно уместить на этих страницах. Сейчас же я просто к великому Марио добавлю еще один пример.

Жанр RPG зародился в мире настольных игр. На экраны мониторов он перекочевал в лице Wizardry и Ultima, но это были серьезные игры для взрослых высоколобых ребят. Идеи этих сложных и уникальных проектов, а также опыт коллег, создавших Dragon Quest, легли в основу самой первой части **Final Fantasy**. Именно FF популяризовала жанр JRPG на консолях и доказала, что японские игры могут быть интересны и западной аудитории всех возрастов, а жанр RPG не является нишевым и малопонятным развлечением.

Нужно целиком и полностью абстрагироваться от массовой культуры, чтобы ни разу в жизни не слышать имени Марио или названия Final Fantasy. Такие титаны не рождались уже очень давно. И можно было бы парировать тем аргументом, что в восьмидесятые создавалось не так много игр и выстоять в конкурентной борьбе с коллегами было проще, а значит, было проще и выделиться, но стоит иметь в виду, что до выхода Марио и Final Fantasy игровую индустрию уже хоронили: на рынке был переизбыток видеоигр. Тиражи нашумевшей **E.T.** для Atari утилизировали путем буквального захоронения в земле, а в успех приставки NES на западном рынке до конца не верил никто, так как Запад на тот момент уже «устал» от видеоигр после наплыва низкокачественных продуктов на приставки предыдущего поколения.

Позже, перед самым выходом консоли Nintendo64, президент компании Nintendo Хироси Ямаути говорил, что «рынок уже сейчас зашел в тупик. Он завален новыми играми, 70 % которых не находят своих поклонников». Это высказывание было сделано в 1996 году. Игровую индустрию никогда не переставали считать перегруженной новыми проектами и мантру об избыточном количестве игр всю дорогу повторяют в средствах массовой информации и в наши дни.

Подобные беспочвенные утверждения не являются поводом обесценивать заслуги разработчиков прошлого, как и не должны стать причиной опускать руки перед выходом на «перенасыщенный» рынок в наши дни.

Рынок видеоигр абсолютно всегда называли перенасыщенным, но это ничего не значит. Здесь всегда есть место новым именам. В том числе и вашему.

Условия, в которых выходили на Западе Final Fantasy и Super Mario, были, напротив, гораздо катастрофичнее, чем условия для выпуска игр

в наши дни: игровое сообщество не отличалось многочисленностью; количество проданных приставок не составляло десятков миллионов; игру приходилось продавать строго на физических носителях, в то время как сейчас вы можете распространять любое свое творение через Интернет, не заморачиваясь с поставками, логистикой и производством картриджей. В наши дни вы можете даже выпустить игру с ошибками, глюками и багами, а через какое-то время залатать все недоработки «патчем», скачивание которого многие пользователи даже не заметят. В былые же времена обнаружение серьезного бага в игре могло привести к отмене поставки картриджей, а значит, к колоссальным финансовым потерям.

Не только игровой рынок был негостеприимным для творчества: сами инструменты для разработки и жесткие требования к разработчикам создавали воистину спартанские условия для творцов.

Узнаваемый стиль Super Mario создавался с использованием 64 цветов; битвы с ордой монстров в Final Fantasy умещались на экране 256 × 240 пикселей. Ограничения в количестве возможных для отображения цветов вынуждали разработчиков FF использовать всего лишь три цвета на каждого отдельного персонажа, а часть монстров была раскрашена с использованием одинаковой палитры – только так NES могла отобразить их на одном поле битвы.

Отдельные двухмерные объекты на экране называются «спрайтами». В понятие **«спрайт»** входили и интерактивные объекты, и персонаж, и враги, и элементы окружения вроде кустов и деревьев. Любая отдельная двухмерная картинка у вас на экране – это спрайт. Спрайт может быть как анимированным, так и статичным. Если вы нарисуете в какой-нибудь программе дерево на прозрачном фоне и вставите это дерево в игру, то дерево смело можно назвать «спрайтом».

Чтобы понять, в каких суровых ограничениях работали Миямото и его команда, можно взглянуть на всю графику в Super Mario *(рис. 1)*.

Рис. 1. ***Вся графика из игры Super Mario Bros., NES, 1985 год.***

Почему же мы не видим здесь отдельного спрайта для Марио, отдельного спрайта для боссов и для окружения? Где облака, где кусты? Что это за каша такая? А дело в том, что на NES крупные спрайты состояли из множества квадратиков 8 × 8 пикселей, каждый из которых имел несколько вариаций, чтобы создать анимацию. Спрайт Марио складывался из двух квадратиков, последовательная смена которых создавала иллюзию движения.

Художники не могли просто нарисовать Марио во весь рост и отправить его в игру в таком виде.

Раскрашивались спрайты программно, потому в игру они вносились в черно-белом варианте. Это позволяло использовать один и тот же спрайт, например, для облаков и кустов достаточно было облако покрасить в белый цвет, а куст – в зеленый.

Модуль обработки изображений NES отрисовывал всего два слоя. На верхнем из них располагались спрайты. Отдельный спрайт не мог быть больше, чем квадратик 8 × 8 пикселей, и для достижения такой цели, как «создать крупный спрайт персонажа», художникам приходилось разбивать изображение на 4, 6 или еще большее количество квадратиков. Анимации для каждого квадратика создавались отдельно.

Не менее сложный подход распространялся и на дизайн уровней. Второй из слоев, отображаемых модулем обработки изображений, был фоном. Фон состоял из 900 плиток разрешением 8 × 8 пикселей, но лишь 256 из них могли быть уникальными, в то время как остальные 644 обязаны были быть их копией. Меньше одной четверти экрана могло быть занято уникальными объектами!

Хоть приставка и могла обрабатывать 256 цветов, фону конкретно в Final Fantasy отдавалось 13 цветов, один из которых и без того был занят базовой заливкой. Оставшиеся 12 цветов разбивались на четыре палитры, но и на этом ограничения не заканчивались: система объединяла каждые 4 плитки 8 × 8 в один блок, и использовать в этом блоке можно было только одну из заготовленных палитр.

Файл со всей графикой в Марио занимал 8 Кб памяти. Чтобы так умело распределить объекты и разбить все на кубики 8 × 8, художник был обязан мыслить так, как мыслит программист. Здесь не шло речи о «свободе творчества» и непринужденных мазках кистью. Для создания подобной графики не хватит одних лишь академических навыков классического рисования. Чтобы сделать картинку выразительной, читаемой, узнаваемой и красивой, требовалось множество умений и знаний из областей, не относящихся к изобразительному искусству.

Массивное количество суровых ограничений превращало работу над визуальным стилем в изнурительную игру в невероятно усложненную версию шахмат. Разработка игр в таких сковывающих условиях в наши дни кажется абсурдной и невыполнимой задачей *(рис. 2)*.

Можно углубиться еще дальше в системные ограничения NES и ужаснуться тому, что весь код игры приходилось писать на низкоуровневом языке **Assembler**, внешний вид которого способен ввести в состояние ужаса любого современного программиста. Командам Миямото (Super Mario) и Сакагути (Final Fantasy) приходилось писать свой собственный рендер; качество графики в их играх зависело не только от художников, но и от программистов; ребята не могли даже поворачивать спрайты! В NES не было такой функции! Если вы видели в играх крутящийся спрайт, то такой эффект был достигнут рядом изобретательных и оригинальных решений, а не простой современной командой rotate to angle. А про всего лишь пять голосовых потоков, которые были доступны авторам музыки для NES, можно написать отдельную книгу – и она будет достойна внимания,

ибо в эпоху NES начало зарождаться такое направление, как chiptune – «восьмибитная» музыка, у которой все еще есть поклонники и авторы.

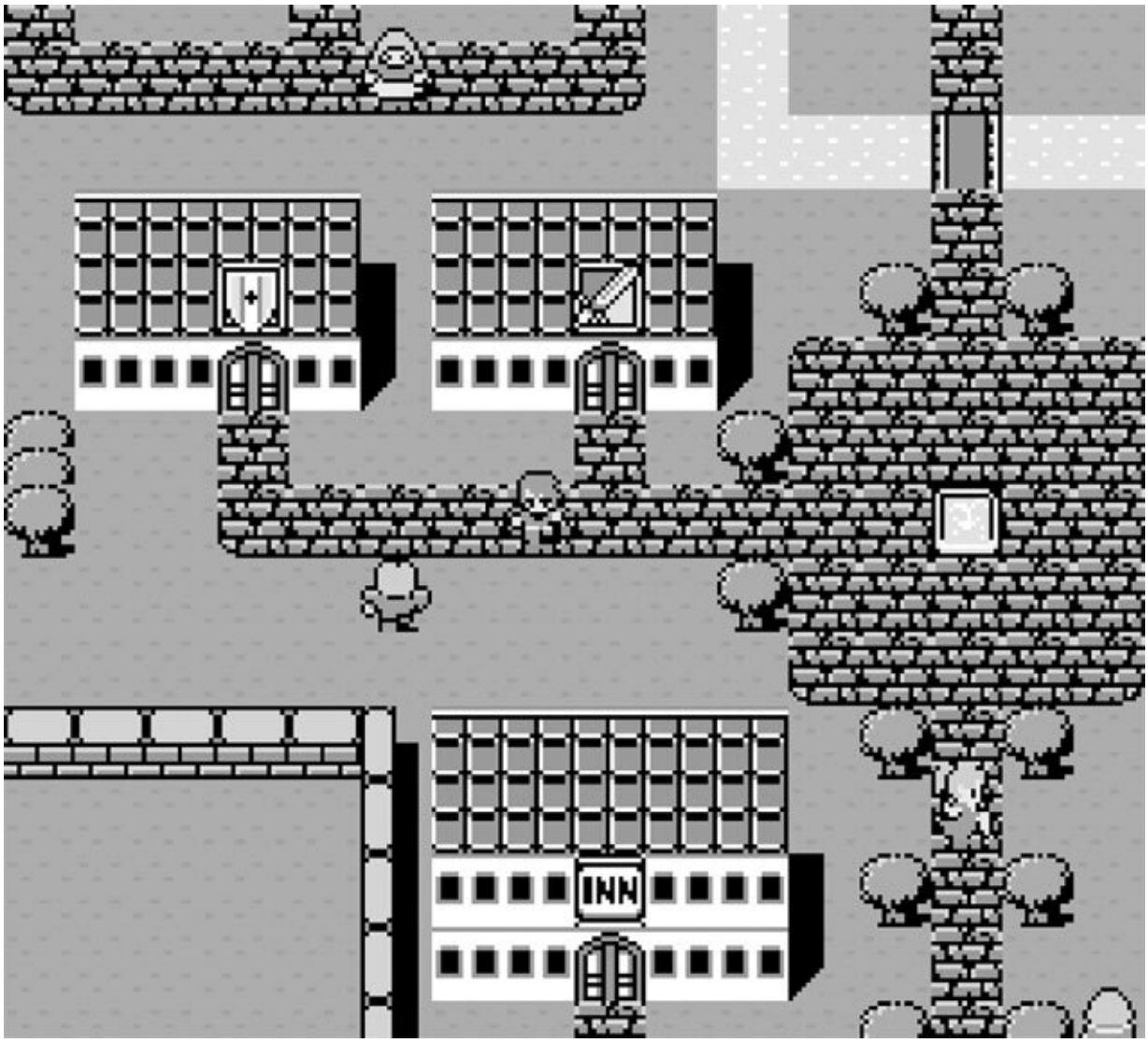

Рис. 2. ***Final Fantasy, NES, 1997 год***

Команды и Миямото, и Сакагути состояли всего из четырех человек. Четыре человека, находясь во внушительных ограничениях, обладая, казалось бы, неимоверно скудными возможностями, смогли создать нечто столь грандиозное и революционное, как Super Mario и Final Fantasy.

В восьмидесятые не было Youtube с его колоссальным количеством уроков. Не было книг и курсов. Не существовало понятия pixel-artist (тот, кто рисует изображения с помощью «пикселей» – самых крошечных объектов на экране), а ведь рисовать на уровне пикселей – это отнюдь не то же самое, что заниматься классическим изобразительным искусством. Разработчики не могли перенять опыт своих коллег, потому что со своими революционными идеями смелые и умные ребята выходили на абсолютно не истоптанную тропу, где они оставались со своими уникальными проблемами один на один.

Но что самое интересное – у них получилось создать нечто волшебное, удивительное, положившее начало целым культурам и вырастившее не одно поколение игроков. У них не было и десятой части тех инструментов, что есть сейчас у нас. Наши возможности – гораздо шире, чем были у разработчиков в конце восьмидесятых, и сделать мы можем в разы больше. Свою Final Fantasy у вас сейчас получится создать играючи.

> **Уделите внимание ретро-играм эпохи NES, Snes и Sega Mega Drive и сравните их с современными независимыми играми. Не собраны ли, на ваш взгляд, современные игры из точно тех же элементов, из которых создавали хиты прошлых лет?**

Давайте поговорим о том, что именно мы имеем на текущий момент.

https://t.me/it_boooks

2. Идеальное время для одиночек: движки

Начнем с хороших новостей: вам больше не нужно изучать Assembler, чтобы сделать игру. Более того, можно вообще не знать никаких языков программирования и не получать образования программиста. Я, например, так и не удосужился этого сделать. По образованию я психолог.

Для разработки видеоигр в наши дни используются «игровые движки», коих развелось великое множество, но упоминания достойны единицы.

Игровой движок представляет собой программное обеспечение, запуск которого откроет вам такие возможности, которые не снились ни авторам Super Mario, ни авторам Final Fantasy. Представляете – в наши дни с помощью абсолютно любого современного движка можно крутить спрайты! Разработчики игр на NES и Sega MD о таком и мечтать не могли! А знаете, в каких современных движках сохранилось ограничение на вывод всего лишь 64 цветов на экран? Ни в каких! Такого ограничения больше не существует! Вы способны использовать все цвета, которые могут отобразить современные мониторы, а также сопроводить все это дело любыми звуками, которые вам приглянутся, – достаточно просто закинуть. wav или. mp3 файл в игру и не заморачиваться больше о пяти доступных каналах. Вам не нужно собирать спрайты из квадратиков 8 × 8 – вы можете нарисовать спрайт абсолютно любого размера и в любом соотношении сторон.

Перечисление таких возможностей в качестве «удивительных» может звучать саркастично и глумливо, но я отнюдь не ставлю перед собой цель высмеять инструменты прошлого. Я лишь хочу подчеркнуть, насколько доступнее и проще стала разработка видеоигр. Final Fantasy была создана, как уже говорилось, усилиями четырех человек, 3,5 из которых занимались тем, что за вас сейчас сделают движки и программы. Больше не нужно писать рендеры, больше не нужно пытаться уместить всю графику игры в смехотворные 8 Кб памяти. Шансов создать великолепное произведение у нас сейчас гораздо больше, чем было в свое время у Миямото и Сакагучи. Пока

эти ребята двигались на ржавых телегах, мы сейчас будем выбирать свой истребитель среди десятка игровых движков.

«Лучшего» движка не существует. Каждый из них хорош по-своему. Потому сначала стоит хотя бы в общих чертах определиться с тем, что собой будет представлять ваш будущий проект. Будете ли вы использовать двухмерную или трехмерную графику? Планируете ли вы делать шутер, платформер или визуальную новеллу? Говорят, что ответов на эти вопросы хватит, чтобы выбрать себе игровой движок. На самом деле их хватит, чтобы лишь присмотреться к движкам, а самый главный вопрос мы зададим себе, когда поймем, какие движки вообще существуют и чем они отличаются друг от друга.

Первый актуальный движок, на который, возможно, упадет ваш взор, – это Unity. С помощью этого инструмента были созданы такие игры, как Genshin Impact, Outlast и Cuphead. Он бесплатен для независимых разработчиков и, что важно, подходит для создания как двухмерных, так и трехмерных игр. Качество графики, которое вы получите, полностью зависит от вашего упорства – Unity ничем не уступает другим движкам по возможностям выдавать детализированное изображение. Но достоинства движка играют не такую важную роль, как ответ на вопрос «а может ли в нем разобраться разработчик, не обремененный лишними знаниями о программировании?».

Unity поддерживает C#, и знание C# – огромный плюс, который ускорит обучение этому изумительному инструменту. По сравнению с Assembler, на котором создавались шедевры ушедших эпох, C# является весьма высокоуровневым языком, особенно с учетом того, что сам движок постоянно пытается угадать, что мы хотим написать, и пестрит красочными подчеркиваниями, автоисправлениями и предложениями по «правописанию». Тем не менее в наши дни даже C# кажется суровым инструментом высоколобых программистов, в то время как как те, кто не разбирается в C#, могут глянуть в сторону «визуального программирования» *(рис. 3)*.

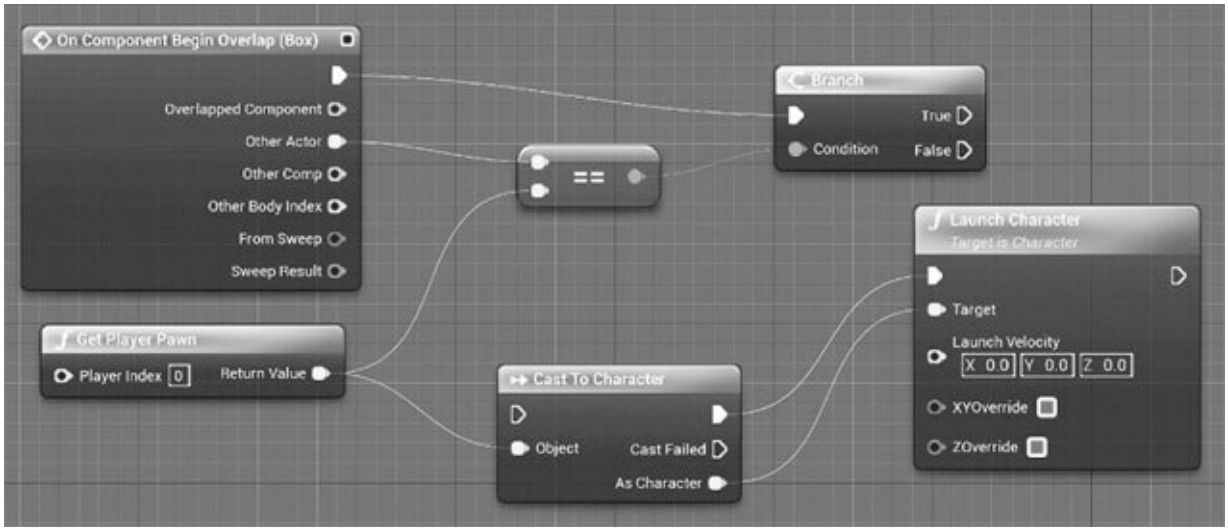

Рис. 3. ***Система Blue Prints в Unreal Engine***

«Визуальное программирование» представляет собой не тот изнурительный процесс, в ходе которого вы пишете сложный код, используя выражения по памяти, а потом ищете, где вы пропустили закрывающую скобку или очередную маленькую запятую. «Визуальное программирование», несмотря на красивое и сложное название, являет собой перетягивание окошек и натягивание стрелочек. Для освоения «визуального программирования» не нужны знания, которые придется получать несколько лет в университете: чтобы совладать с этим «макаронным монстром», достаточно научиться понимать логику движка. Используя свойственный движку «метод мышления», вы будете «объяснять» ему, как должна работать ваша игра. Даже запоминать никаких выражений не придется: при создании «события» вы всегда будете видеть список возможных условий.

Я – не программист. C+, C# и Phyton для меня являются трудноотличимыми друг от друга иероглифами. Тем не менее моим основным заработком является разработка видеоигр, и три выпущенные игры сейчас полностью обеспечивают мое существование. Удалось мне это благодаря «визуальному программированию», которое легко освоить, даже не обладая техническим складом ума.

Мне часто приходится сталкиваться с критикой моего убеждения, что «визуальное программирование» – это легко и просто, а мои слова

принижают таланты людей, которые создают игры, используя тот же инструмент, что использую я. Все, что я могу порекомендовать критикам, – это почитать еще раз о том, в каких условиях и какими инструментами создавалась Final Fantasy, и тогда сразу станет понятно, чья работа представляла собой вечный поиск оригинальных решений, а чья – больше напоминает прогулку по парку в теплый летний день.

Не нужно выстраивать вокруг себя барьер из убеждений, что вы занимаетесь сложной и изматывающей работой. Напротив, чем сильнее ваша «работа» будет ассоциироваться у вас с чем-то легким и приятным, тем больше вероятность, что и утомляться вы будете меньше. Если вы научились получать от работы в движке удовольствие – возрастет тяга к тому, чтобы сесть наконец-то за рабочее место и начать делать игры.

К психологическим барьерам мы еще вернемся – других же барьеров перед нами на самом деле не выстроено. Время снова поговорить про Unity, который гордо выставлял грудь вперед на протяжении нескольких абзацев текста, а сейчас делает огромный шаг назад. Хоть инструменты для визуального программирования на Unity и существуют (один из них называется **Bolt**), они не являют собой основной способ взаимодействия с движком и развиваются скорее «параллельно ему». Помимо Bolt есть множество других надстроек на Unity, с которыми можно ознакомиться здесь по ссылке на *рис. 4*.

Каждый из них формирует внутри движка свою собственную среду для создания игр определенного плана: платформеров, новелл, шутеров. Помимо удобного инструментария эти надстройки «награждают» разработчика неповоротливостью и неспособностью использовать функционал Unity на полную мощь, вынуждая творца все-таки прибегнуть к изучению хотя бы каких-то аспектов C# в случае, когда его воображение выйдет за рамки, которые создают подобные надстройки.

https://assetstore.unity.com/tools/visual-scripting
Рис. 4.

Вторым титаном среди игровых движков выступает **Unreal Engine**, блистающий своей технологичностью и всегда словно опережающий свое время. На нем были созданы Fortnite, PUBG и Hellblade. Распространяется этот мощный и серьезный движок бесплатно, но в случае если ваши доходы превысят определенное значение, команда Epic Games попросит вас оплатить Royalty – процент со своего дохода. Этим «определенным значением» сейчас выступает миллион долларов, и я искренне буду рад за тех, кому придется платить Royalty, – ведь это означает, что ваша игра стала чрезвычайно успешна.

Я не побоюсь утверждать, что именно Unreal Engine стоит за популяризацией визуального программирования: его система Blueprints уже интегрирована в движок и доступна для понимания каждому, вне зависимости от навыков и склада ума. Epic Games щедро предоставляет своим пользователям доступ к куче исходников, которые можно использовать как обучающие материалы. Например, если вы собрались делать многопользовательскую игру, то исходник **Lyra** для Unreal Engine 5 сможет стать неплохой основой для воплощения в жизнь такой, казалось бы, трудной задачи.

Кругом витает убеждение, что использование любой технологии может стать ошибкой, если технология эта применена не в той отрасли. Так, например, создание 2D-игры на Unreal Engine для многих не выглядит разумным решением: этот могучий инструмент абсолютно не предназначен для работы с плоскими картинками. Вы же не будете использовать самолет для того, чтобы добраться в «магазин-через-дорогу», верно? Ровно как и использовать велосипед для того, чтобы перебраться на другой континент, вроде бы неразумно, но именно так для многих выглядит попытка создать нечто высокотехнологичное на движках **RPGMaker, Construct, Gamemaker,**

Godot или **Ren-py**. Все они предназначены для работы только с 2D-графикой, а Ren-py так и вовсе разработан только для создания визуальных новелл.

3. Движки и ваши личностные особенности

Я только что рассказал про два могущественных движка, которые позволяют воплотить в жизнь любые по сложности идеи, но тем не менее на Unity и Unreal Engine создается лишь около половины игр, выпускаемых в Steam. Некоторые крупные AAA-студии продолжают использовать свои собственные закрытые движки, чтобы меньше зависеть от сторонних компаний, и к этим ребятам вопросов у нас не имеется. Но что же движет небольшими студиями и соло-разработчиками, которые выбирают себе в качестве основного инструмента другие, куда менее популярные и продвинутые продукты?

Unreal Engine и Unity, по сути, не сложнее в освоении, чем другие «мелкие» и «неповоротливые» движки. С одной стороны, научиться на них работать даже проще: у Unity есть официальный бесплатный курс для начинающих разработчиков, а уж количество обучающих материалов по Unreal Engine зашкаливает, в то время как найти толковый обучающий курс по какому-нибудь Construct – задача непосильная.

Люди, выбравшие инструментом для воплощения своих идей весьма примитивные программы, на мой взгляд, опровергают популярное убеждение, упомянутое мною выше: якобы самое важное – чтобы инструмент использовался по назначению.

Но главное в выборе движка – это ваши личностные особенности. От них-то и надо отталкиваться.

Если вам будет некомфортно работать на выбранном движке, то вы не будете на нем работать. При разработке игры в одиночку нам ничто не мешает опустить руки и бросить свои светлые начинания. Нам важно создать условия, в которых вероятность устать от создания игр будет минимальной.

Ключевым нюансом в выборе движка является то, насколько вам приятно и удобно с ним работать. Уделите по одному вечеру каждому из движков, выполните в них по

одному уроку и определитесь с тем, где вам было приятнее работать.

Даже если вы собрались делать 2D-игру, но вас привлекают эстетика «высокотехнологичности», сияющая новизна и чувство причастности к чему-то очень продвинутому – выбирайте Unreal Engine. Если же вам неприятно работать в Unreal Engine и он кажется вам излишне перегруженным, то какой бы проект вы ни делали с помощью этого инструмента – он навсегда останется недоделанным.

Если у вас есть планы найти работу в среде разработчиков, делать мобильные приложения или устроиться в игровую студию, то Unity на данный момент является движком, навык владения которым быстрее всех поможет вам найти работу. Если же вы все время спотыкаетесь о C#, чувствуете, как написание кода вас тормозит и мешает воплощать идеи в жизнь, а использование 3D для имитации 2D кажется вам глупостью, то у вас не получится раз за разом возвращаться к интерфейсу Unity и создавать свой продукт.

Construct – движок для работы только с 2D-графикой. Именно он – мой личный выбор. Construct полностью основан на визуальном программировании и создает среду, в которой работа с неким подобием кода протекает невероятно быстро. Я человек абсолютно не технического склада ума. Меня не пугает, что я не могу заглянуть «под капот» движка и до конца понять все процессы, которые скрываются за аккуратными «блоками» с условиями. Мне это просто неинтересно. Я могу уделить больше внимания спрайтам и визуальной составляющей моей игры. Именно внешний вид создаваемых мной продуктов для меня является важнейшим аспектом: я одержим контролем над каждым спрайтом, над каждой тенью и каждым эффектом, оттого предпочитаю классическую анимацию и полное отсутствие программных спецэффектов и постобработки в своих проектах. Такие игры, как **Hollow Knight**, **Cuphead** и **Hotline Miami**, не были созданы на Construct2, но могли бы – движок позволяет создать все то, что мы видели в этих играх *(рис. 5)*.

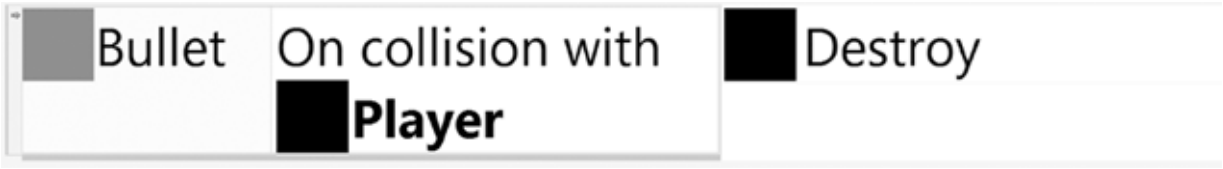

Рис. 5. ***Логика поведений в Construct. Слева – условие (пуля коснулась игрока), справа – событие (игрок уничтожен)***

Упомянутая Hotline Miami была создана с помощью **Gamemaker**. Точно так же как и Construct, этот движок запирает разработчика в двух плоскостях, лишая его возможности создавать трехмерные игры. В Gamemaker используется собственный язык GML, который имеет и визуальное ответвление. GML отличается от языка Construct своей глубиной и сложностью, что делает Gamemaker инструментом, подходящим для людей с техническим складом ума. Если вас воодушевляет и дает силы работа именно с кодом, но при этом профильного образования у вас пока еще нет, то в создание игр на Gamemaker вполне можно втянуться.

Мне бы очень хотелось составить табличку в духе: если вам от движка нужно вот «это» и «это» – выбирайте «такой-то движок», но, во-первых, такие таблицы уже лежат на просторах Сети, а во-вторых (если вы правильно уловили мою мысль), существование такой таблицы не очень поможет вам выбрать инструмент.

Если при выборе движка отталкиваться только от технических требований к своей будущей игре, то велика вероятность, что вы натолкнетесь на программу, с которой не сможете «подружиться» и в среде, в которой не будете чувствовать себя комфортно. Если бы сотни моих вечеров, проведенных за работой в Construct, являли собой борьбу с логикой движка, его интерфейсом, а заодно с моим дискомфортом и ярым нежеланием возвращаться к работе, то я бы не довел до ума ни один из своих проектов. Именно соответствие Construct моим личностным особенностям помогло мне создать на нем три игры и провести в этой программе больше 6000 часов.

Работа с игровым движком напоминает общение между двумя людьми из разных стран. Вы очень плохо знаете язык приезжего, а если и заучите каждое слово, то вам все равно потребуются дополнительные годы, чтобы говорить на уровне носителя языка и понимать все культурные особенности своего товарища.

Каждый вечер вы будете пытаться объяснить этому иностранцу на ломаном подобии его наречия: «Я хочу, чтобы ты сделал “вот такую штуку”» (например, чтобы персонаж атаковал по нажатию на кнопку «X»). Чтобы в понятной форме донести до движка ваши требования, вы должны быть терпеливы и внимательны. Сложно оставаться терпеливым и внимательным к продукту, который вызывает у вас только раздражение и ненависть.

Один мой знакомый был программистом веб-сайтов и неплохо знал такой язык программирования, как Java Script. Желание делать игры привело его к Unity – благодаря многочисленному сообществу и отличной маркетинговой кампании пути многих начинающих разработчиков упираются именно в Unity. В ходе разработки своего проекта с использованием всех предоставляемых Unity благ он начал спотыкаться о реализацию сложных и глубоких задач, вроде создания AI (искусственного интеллекта) или проработки поведения NPC. Знания Java Script едва ли помогали ему в этом деле. Скорее, напротив, они мешали освоить новую логику в новой среде.

Вместо удобного, раскрученного, многофункционального движка мой знакомый в итоге обратился к более низкоуровневому программированию и начал разработку проекта на «сухом» Java Script с использованием react-компонентов (готовой библиотекой некоторых шаблонов кода для реализации конкретных задач) и массово используемым готовым решением для корректно работающей физики под названием phaser.js. Работа над игрой стала для него куда менее утомительной, среда – куда более понятной и знакомой лично ему. Но набор его инструментов выглядит как что-то неповоротливое и сложное! Так почему ему не понравилось работать на Unity?

Его личный опыт, его склад ума и его знания поспособствовали тому, что работа на чем-то более, казалось бы, сложном и неудобном, протекала быстрее и приносила больше удовольствия.

Надеюсь, никто не забыл предисловия к этой книге, и не ждет, что я четко скажу вам в одном предложении, какой движок вам выбрать? Я не могу этого сделать – проблема выбора движка лежит на вас, я уже перечислил, на что нужно опираться в своем решении. Но я могу дать еще две подсказки.

Во-первых, никто не запрещает вам попробовать каждый из популярных движков. Зайдите на сайт одного из них, посмотрите на

игры, которые созданы с его помощью, взгляните на пару вводных уроков и прислушайтесь не к своему разуму, а к более глубоким ощущениям – цепляет ли вас хоть чем-нибудь эта программа? А продукты, созданные на ней? У некоторых игр на Unity есть некое неуловимое сходство, отличающее их от игр на Unreal. А какое ощущение оставляет беглый просмотр форума сообщества? Вы почувствуете, как что-то внутри в определенный момент щелкнет и даст ответ на вопрос: «А для меня ли эта программа?»

Если прислушаться к себе окажется чрезвычайно трудной задачей, то скачивайте все подряд движки и пытайтесь в каждом из них выполнить хотя бы один урок. Так вы точно поймете, выполнение какого урока принесло вам удовольствие и на каком движке стоит остановиться. Без удовольствия от работы вы ни за что не заставите себя провести месяцы, а может быть, и годы свободного времени за созданием видеоигр.

Второй совет – общайтесь с разработчиками. Никто лучше человека, работающего на Game Maker, не ответит вам на вопрос «а легко ли на нем будет сделать “вот-такую-то-игру”»

Лучше, конечно, общаться в реальной жизни – так вы уловите больше их личных свойств и характерных манер.

На мой взгляд, существуют некоторые неуловимые качества, объединяющие тех, кто выбрал себе в «коллеги» тот или иной движок. Эти незримые сходства и трудноуловимые качества и должны стать еще одним вашим помощником в выборе движка – кто из разработчиков окажется ближе к вам по своим манерам, взглядам и убеждениям?

• Мы не одни

https://t.me/it_boooks

Поиск живого общения с будущими коллегами по цеху должен увенчаться прекрасным открытием: во множестве городов России регулярно проводятся как формальные, так и неформальные сходки разработчиков. Если их не бывает в вашем городе, то помните, что поезда все еще ходят и, потряхивая ваши косточки в купе или плацкарте, доставят вас до места встречи разработчиков.

Наши сходки условно можно разделить на три формата.

Первый из них является самым легким и непринужденным и подразумевает встречу в специально арендованном зале какого-нибудь бара, где можно пересаживаться из-за одного столика за другой и коротать время в беседах с людьми из игровой индустрии.

Второй формат – чуть более деловой, и помимо необузданного и хаотичного общения он подразумевает наличие в программе доклада от одного из участников сходки. Во время выступления местный разработчик будет рассказывать про свой опыт, отвечать на вопросы и делиться полезной информацией.

Если вы живете в Петербурге или планируете его посетить, то поиск мероприятий должен привести вас к **«Индикатору»** – площадке для разработчиков видеоигр, где лекции сочетаются с необузданным общением. Чтобы только послушать доклады, стоит обратить внимание на сообщество игровых разработчиков **Braindie**, а провести время в совсем уж легкой обстановке можно с **GamedevHouse** – сообществом, которое организует сходки еще и в Москве. В поисках площадок, где читают лекции для разработчиков игр в столице, можно присмотреться к мероприятиям от **ВШБИ**, а гостям и жителям других городов – прибегнуть к помощи Интернета и социальных сетей, потому что я, разумеется, не могу перечислить все мероприятия во всех городах.

Третий формат, самый, казалось бы, серьезный из всех, – это конференции. Одно слово «конференция» рисует перед глазами серьезную встречу угрюмых дядечек в пиджаках, которые говорят только о будущих сделках, а за могучими плечами каждого из них стоит как минимум одна AAA-игра.

Я же предпочитаю описывать конференции как «бизнес-праздники», в которых «бизнес» вполне можно отодвинуть на задний план. Чаще всего конференция проходит в красочно обставленном зале, где и крупные компании, и маленькие независимые разработчики вроде нас демонстрируют свои проекты, участвуют в конкурсах, ищут коллег, а главное – общаются, делятся опытом и весело проводят время.

Из крупных российских конференций необходимо упомянуть **White Nights**, организаторы которой регулярно проводят еще и серии лекций как раз для тех, кто пришел на конференцию не как журналист, издатель или разработчик, а ищет новых знаний для того, чтобы приобщиться к индустрии.

Причины посещать эти мероприятия кроются далеко не в контактах, связях и знакомствах, без которых, честно говоря, разработчик-одиночка теоретически может обойтись. Как и все самое важное, причины приехать в другой город на сходку или конференцию кроются у нас в голове.

Мы все – социальные животные. Общение и социализация для нас жизненно необходимы. Даже если мы работаем в гордом одиночестве, потребность в общении никуда не пропадает. Ее не обязательно чувствовать, но поверьте – такая потребность существует.

Я уверен, что на свете есть индивиды, которые действительно способны прожить счастливую жизнь без взаимодействия с другими людьми, но с очень малой вероятностью именно вы являетесь одним из них. Когда каждый на этом свете считает себя интровертом (а уж человек, который избрал путь разработчика-одиночки, скорее всего убежден в таком качестве своей натуры), я чувствую на себе ответственность за рассказ про некоторые вещи, скрытые внутри нашей черепной коробки.

Во-первых, ваши социальные инстинкты будут постоянно требовать от вас быть «подключенным» к стае. Такая механика работы нашего мозга выработалась в связи с тем, что в первобытные времена в одиночку человеку выжить было крайне сложно. Те, у кого отсутствовала зудящая потребность стать частью общества, пытались в одиночку охотиться, строить себе жилье и обеспечивать себе безопасность. Совершенно очевидно, почему они не смогли передать свои инстинкты и свой подход к существованию дальше, – такие люди

просто умерли, потерпев неудачу в своем начинании выжить в одиночку.

Десятки тысяч поколений передавали друг другу тягу к объединению в стаю не для того, чтобы современный человек решил, что он «интроверт» и попытался существовать, ни с кем не общаясь. За десятки тысяч лет психика человека не изменилась никаким образом. Наш мозг все так же адаптирован на жизнь в пещере, а не в мегаполисе.

Сейчас мы можем не ехать на сходки в другие города – мы можем пообщаться с другими разработчиками в Сети, не подозревая о том, что это – всего лишь иллюзия настоящих разговоров. Во время живого общения с людьми у вас задействуются совсем иные области мозга, нежели те, которые мы используем при анализе информации в Интернете. (Орбитофронтальная кора активизируется, когда вы общаетесь с настоящими людьми, а височные доли – когда вы взаимодействуете с экраном телефона.)

Тут возникает парадокс: вам кажется, что вы общаетесь с людьми достаточно много – в ходе переписки или просмотра роликов в Интернете. Но никакого удовлетворения социальных инстинктов такое «общение» вам не принесло. Мы подвергаем свою эмоциональную стабильность огромному риску, изолировавшись от других людей за экраном телефона.

Мозг любого из нас не приспособлен к существованию в изоляции. Если мы не будем чувствовать, что мы – часть чего-то большего, то социальный инстинкт так и останется неудовлетворенным. Вы, скорее всего, этого даже не осознаете: мозг просто без вашего ведома решит, что вы несчастны; химические процессы в голове поменяются, дофамина и серотонина станет вырабатываться меньше, у вас начнутся **депрессия** и **упадок сил**. Такое унылое состояние очень пагубно скажется на способности разработать игру в одиночку, а мозг так и не сообщит вам: «А давай-ка махнем на сходку!» – вам нужно догадаться, каков выход из сложившейся проблемы, самостоятельно.

> **Посетите сходку разработчиков. Почувствуйте, что вы часть чего-то большего. Это необходимо для удовлетворения ваших социальных потребностей.**

Вторая скрытая причина, по которой сходки разработчиков сто́ят того, чтобы выбираться из уютной квартирки и ехать в далекие дали, заключается в том, что общение с другими людьми формирует в нашей голове некие образы.

Мы не способны познать ни одного человека до самого конца, до всех мелких деталей его мышления. Нам может казаться, что мы чудесно понимаем людей, осознаём все, что они нам говорят, и улавливаем смысл каждого их действия. Только стоит задуматься: а существует ли на свете кто-либо, кто на 100 % понимает нас самих?

Так если нет ни одного человека, который понимает нас на 100 %, то как мы можем быть уверенными, что существуют люди, которых мы поняли на 100 %?

При общении с людьми формируются и обрастают деталями их образы у нас в голове. Именно эти образы, скомпонованные нами самостоятельно, мы и можем понять. Как сильно отличается этот образ от реального человека, зависит от нашей внимательности и особенностей восприятия, но образ этот никогда не будет полностью соответствовать самому человеку. Особенно если знакомство с ним поверхностно.

Наше представление о других людях никогда не соответствует действительности. Наше представление о них – это то, как мы их воспринимаем.

На сходках разработчиков формируются именно поверхностные образы, но они-то нам и нужны! Мы сами наделим образы разработчиков бо́льшим количеством личностных качеств, которые будем потом трактовать как качества, необходимые для разработки успешных игр.

На мероприятиях я нередко встречаю людей, творчеством которых я восхищаюсь. Когда мне удается пообщаться с разработчиком, который был причастен к созданию игры, что мне полюбилась, это формирует в моей голове образ человека, достигшего того, чего хотел бы достичь я. Тут и приходит важное понимание, что разработчики видеоигр – это далеко не сверхлюди.

Раньше мне казалось, что человек, делающий игры (особенно в одиночку!), представляет собой некое многорукое сверхсущество, которое 20 лет училось рисовать, 20 лет училось писать код и еще 30

лет изучало все тонкости самодисциплины. Такой образ очень сильно отличался от того, каким я видел сам себя.

Мне начинало казаться, что для того чтобы делать игры, я должен очень сильно преобразиться. Перестать был тем, кем я являюсь сейчас. Стать дисциплинированным чудовищем. Наш мозг не особо-то любит перемены и всегда будет стараться избегать того, что угрожает представлению человека о себе. Процесс «избегания» будет состоять из бесконечного поиска причин, почему нам не стоит заниматься разработкой видеоигр. Имея в голове представление о разработчике как о многоруком чудовище, мы вечно будем располагать оправданиями, чтобы ничего не делать: у нас всегда не будет хватать на разработку времени, сил, вдохновения, мотивации или чего-нибудь еще.

Объективной причиной бездействия в нашем случае станет естественное нежелание вашего мозга настолько кардинально менять свое «я» для достижения цели. Даже если разумом вы будете понимать, что приобретение новых качеств и изменение собственного «я» сделает вашу жизнь лучше и краше, ваш мозг продолжит сопротивляться и отказываться работать.

Но вот вы на сходке разработчиков. Или на конференции. И вы видите не многоруких монстров с тремя головами и многолетним стажем за спиной, а обычных молодых ребят, распивающих напитки, экономящих на закуске и отпускающих шутки на отдаленные от разработки темы.

Образы этих людей потеряют ту «монструозность», которая провоцирует ваш мозг принимать защитную позу и воспринимать новое увлечение как угрозу вашей идентичности. Путь от образа «я-сейчас» к образу «я-разработчик» перестанет казаться вашему мозгу длинным, тернистым и ведущим к такому огромному количеству изменений, в ходе которых от вашего нынешнего «я» не останется и следа.

Очеловечивание образа разработчика поможет вам избавиться от некоторых оков, что мешают вам развиваться как профессионалу.

5. Чужой среди своих

Я сталкивался со странным убеждением, что на таких сходках и лекциях стоит появляться только тогда, когда за плечами у вас будет то, что можно показать людям: желательно, уже выпущенная игра, заработавшая миллиард долларов. Требования к себе у людей разные, можно много перечислять причины стесняться и сидеть дома, но у всех этих причин будет одна общая черта: абсурдность.

У меня для вас есть две новости: плохая и хорошая. Плохая новость заключается в том, что человеку, с которым вы начнете беседу на сходке, будет плевать на то, чем вы занимаетесь, даже если за плечами у вас что-то воистину выдающееся. Хорошая новость заключается в выводах из плохой новости: если вы ничего не умеете, никому не будет до этого дела, а значит, вам нечего стесняться своего статуса «новичка».

Разработчики – точно такие же люди, и для каждого из людей самым важным и интересным является одна-единственная вещь – он сам. Каким бы неудачником или мастером вы бы ни были, вашему новому знакомому будет всегда интереснее говорить о себе, а не о вас.

Чтобы показаться отличным собеседником, вам достаточно просто внимательно слушать. Я часто начинаю знакомство на сходке с фразы «привет, а чем ты занимаешься?» В ответ я получаю «я 3D-моделлер», «я музыкант» или «я дизайнер уровней». Это уже дает возможность продолжить беседу, задавая дополнительные вопросы о том, где человек работает, как он научился тому, что умеет, и к каким проектам приложил руку.

Если слушать человека внимательно, давать обратную связь и быть вовлеченным в его речь, то, поверьте, вы не покажетесь скучным, назойливым или приставучим. Напротив, такая беседа останется у человека в воспоминаниях как нечто очень приятное. Позвольте людям самим говорить о себе – и общение с вами станет для них отличным времяпрепровождением.

Не стоит бояться, что в ответ на встречный вопрос «а ты чем занимаешься?» вам будет нечего ответить. Я часто встречаю на сходках людей, которые просто приглядываются к индустрии, но еще в ней не

работают, и я ни разу не видел, как их кто-то унизил за стремление к знаниям.

Два года назад на одном мероприятии со мной познакомился обычный парень, не имеющий никакого отношения к разработке видеоигр, но очень сильно интересующийся тем, как устроена эта отрасль. После знакомства я начал обращать внимание, что он посещает вообще все лекции и мероприятия, которые связаны с созданием видеоигр.

Мы пересеклись с ним еще кучу раз, и с каждым разом он создавал впечатление все более уверенного специалиста в области разработки. Сейчас он делает весьма успешные игры со своей собственной командой – а ведь прошло не так много времени с тех пор, как он стеснялся говорить о себе.

Его пример вдохновляет: в истории этого парня я вижу возможность двигаться вперед без устали и менять свою жизнь согласно собственным желаниям. Помнить такое – это очень вдохновляет. И теперь вы знаете, где и у вас есть возможность приобрести запас вдохновения и заработать новое убеждение. Убеждение, что вы способны делать игры.

Один мой знакомый с восторгом описывал, как поменялось его мышление, когда в гостях у него побывали два профессиональных разработчика. Они сидели у него на кухне, на том же самом месте, где он обычно сидит с женой и обсуждает простые бытовые вопросы. Два его новых товарища говорили про договоры с именитыми издателями и про то, как распорядиться инвестициями. Они обсуждали это так обыденно, словно речь тоже шла о рутинных бытовых проблемах.

Именно присутствие в кругу общения таких ребят помогло моему товарищу сформировать представление о том, что «делать игры – это выполнимая работа». Мы всегда строим планы на жизнь и формируем свою программу действий исходя из того, что нам кажется возможным. Если же в кругу нашего общения одни только люди, профессии которых не связаны ни с риском, ни с творчеством, то вероятность резко сменить профессию на разработчика игр будет куда меньше. Вам и вашему мозгу нужны живые примеры людей, преуспевших в этом деле. Вам нужно общение с разработчиками.

То, о чем вам поведает живой человек, а не запись на YouTube, по многим причинам гораздо лучше отложится в голове. Вместо

посещения доклада от местного разработчика в каком-нибудь баре, безусловно, можно послушать лекцию более именитых деятелей индустрии на YouTube, но толку от этого будет, как ни странно, куда меньше.

Первая лекция, которую я посетил, была посвящена дизайну уровней и проводилась в помещении ночного клуба, работающего в обеденное время как обычный бар. Я все еще учился в аспирантуре в то время. Память об утомительных лекциях в университете и о школьных уроках еще не остыла. Для меня стало приятной неожиданностью, что зубодробительное разгрызание гранита науки в формальных и строгих стенах университета не имело ничего общего с неформальной лекцией в баре.

Отличия заключались в том, что выступающего все слушали с жадным вниманием; никто не скрипел стульями, не залипал в телефонах, не шептался, не хихикал на задних партах. В помещении не было угрюмой атмосферы академической покорности. Общая заинтересованность, страсть к своему делу, светлая атмосфера мероприятия – все это не прочувствовать, если не оказаться на подобном выступлении вживую.

Чтобы сохранить в памяти все услышанное в такой обстановке, не нужно даже особо напрягать мозг. Во-первых, имея в своем распоряжении множество различных маркеров для запоминания, мозг растаскает текст лекций по бóльшему количеству ассоциативных цепочек. Во-вторых, у нас у всех есть еще одна интересная особенность: память человека работает лучше, если он обладает осознанием, что не сможет в любой момент вернуться к этому материалу. Вернуться к лекции в Интернете можно, казалось бы, когда захочется, и мозг сам принимает решение не фиксировать подавляющую часть информации. А вот отмотать время назад и попасть на лекцию, которую вы слушали вживую, – уже невозможно. Мозг воспринимает такую лекцию как «ускользающую» информацию.

Проводились эксперименты, в ходе которых сверялось количество запоминаемой информации студентами двух групп: первая группа имела право только слушать лекцию, а вторая – записывала ее на свои смартфоны любым способом – видео, аудио, что угодно. Весьма очевидно, какая из двух групп запомнила информацию лучше.

Материалы, которых мы не помним, не участвуют в процессе нашего мышления, никак не влияют на нахождение путей решения вставших перед нами проблем и не помогают генерировать новые идеи.

Видео в Сети хороши только в том случае, если вы примените полученные знания здесь и сейчас. В ином случае доклад от местного разработчика мобильных приложений окажется куда полезнее, чем многочасовая лекция с YouTube-канала Game Developer Conference.

Не стоит оставлять без внимания и тот факт, что опыт местного разработчика будет проще применить в собственном проекте. Я слушал про разработку AAA-игр с многомиллионными бюджетами, и это было интересно. Но к чему мне знания о том, как потратить миллион долларов на целый штат сотрудников и выдать целое «ничего», с которым придется клянчить еще миллион долларов у инвесторов? Я не собираюсь таким заниматься!

А если вы все еще сомневаетесь в высокой важности общения даже при работе в одиночку, то подумайте о том, что самое жестокое наказание в тюрьме – это изолятор. Заключенные больше боятся остаться наедине с самими собой, нежели в компании других преступников, среди которых есть по-настоящему грозные и опасные особи.

6. Наши помощники в 3D

По сравнению с восьмидесятыми вместо дьявольского низкоуровневого Assembler в нашем распоряжении находятся удобные движки, работу в которых освоить возможно и без технического образования, а также огромное сообщество разработчиков, где вы обязательно найдете человека, знающего решения всех вставших на вашем пути проблем. Но и на этом наши козыри отнюдь не заканчиваются.

Помимо самого игрового движка можно использовать, разумеется, еще и множество другого программного обеспечения. Иной раз это ПО может заменить целые команды разработчиков.

Так, например, крупные студии пользуются обеспечением **Anima** – оно позволяет создавать толпу управляемых компьютером персонажей, которые будут имитировать настоящую жизнь: прогуливаться по городу, останавливаться, чтобы поболтать, сидеть на скамейках и реагировать на неожиданные вещи. Вы думали, что создатели необъятных открытых игровых миров, где живут сотни NPC, писали код для каждого из них самостоятельно? Как бы не так – это была **Anima** или подобное ей программное обеспечение.

Но речь идет не о таких серьезных и дорогих решениях, а о чем-то попроще.

Если вы собрались делать игру в 3D, то вашим верным другом станет **Blender** – он могуч, он бесплатен, в нем создаются невероятные вещи, а самое главное – на Blender можно установить колоссальное количество дополнений, расширяющих функционал этой программы. Так, например, многим может пригодиться **MB-Lab** – дополнение, позволяющее генерировать людей. Вам не придется прорабатывать каждый полигон человеческого лица и возиться с его «скелетом». Работа в MB-Lab больше напомнит создание персонажа в компьютерной игре, где в нашем распоряжении находятся различные ползунки для регулирования веса персонажа, его возраста и оттенка кожи. Отдельные программы, которые помогут вам сделать 3D-модель человека со всеми костями, – это **MakeHuman** или **Mixamo Fuse**; они представляют собой куда менее перегруженное ПО, нежели Blender, но

и дальше работы с гуманоидоподобными сущностями вы в них на данный момент не продвинетесь.

Для анимирования вашего только что собранного 3D-персонажа вам не нужно прибегать к Motion Capture, который используют для сложных анимаций крупные студии. Сам этот способ подразумевает наличие у вас дорогостоящего оборудования в виде навороченных видеокамер и кучи датчиков, которые вы прилепите к профессиональному актеру и будете заставлять его выполнять различные акробатические трюки. В однокомнатных квартирах этот вариант кажется совсем уж абсурдным.

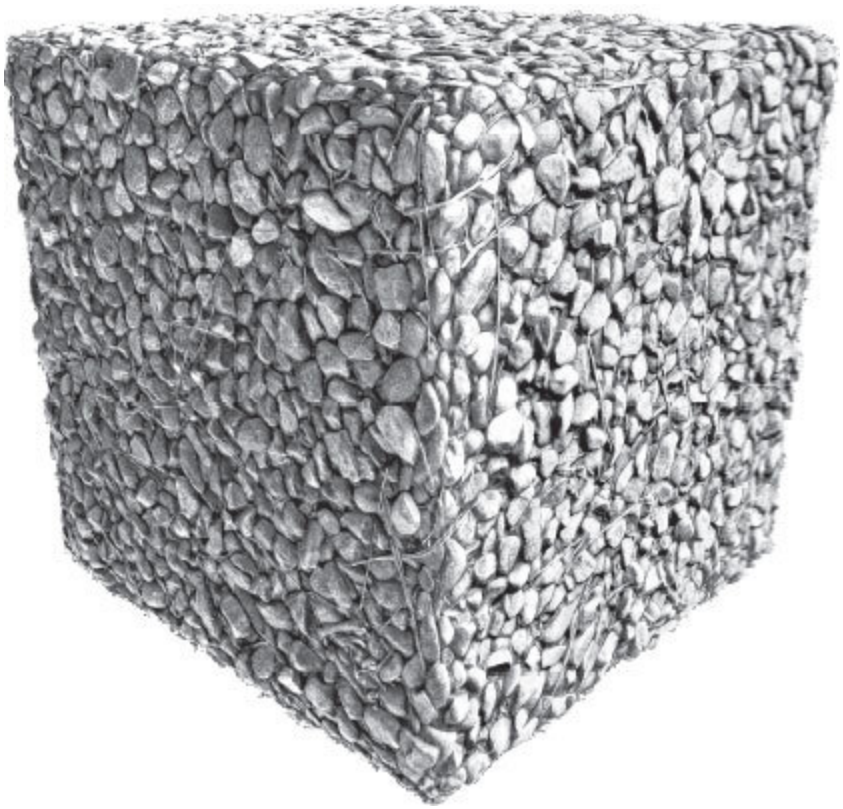

Рис. 6. ***При создании этой модели скульптор не использовал отдельные полигоны для каждого камешка. Это просто куб с наложенной на него PBR-текстурой***

Сейчас и камеры, и датчики, и самого каскадера заменит программа, которая так и называется: **Cascadeur**. С ее помощью вы сможете создавать чрезвычайно реалистичные анимации: программа самостоятельно, без вашего участия, учтет центр тяжести персонажа,

инерцию движения и параболические траектории. Еще более простой путь анимирования 3D-персонажей лежит через онлайн-сервис **Mixamo**, в каталоге которого вы сможете найти невероятное количество уже созданных анимаций, скачать любую понравившуюся лично вам и применить ее на своем персонаже.

Для того чтобы ваш персонаж двигался воистину впечатляюще, пускал молнии, огненные шары, а каждый его шаг поднимал в воздух столбы пыли – можно использовать **Effekseer**. Эта программа являет собой редактор частиц, а ведь именно с помощью частиц проще всего создать подобные визуальные эффекты *(рис. 6)*.

Что касается создания окружения, то и здесь у нас найдутся помощники. Проблемой для начинающего 3D-скульптора может стать не сам процесс моделирования какого-нибудь дерева или колонны, а такой процесс, как текстурирование. Без текстуры ваше дерево будет выглядеть как серый столб, а с текстурой любой серый столб начнет выглядеть как дерево.

Бесплатными помощниками в деле создания текстур могут стать такие программы, как **Laigter, ArmorPaint, Materialize Image To Material Tool** и **Quixel Mixer**. В этих приложениях вы сможете создать PBR-текстуры – они имитируют реально существующие материалы вроде камня, дерева, травы и т. п. Сколы на камнях, отдельные куски коры на деревьях или складки на статичных тканях не всегда в видеоиграх являются действительно многополигональными объектами. Иллюзию высокой степени детализации им создают именно PBR-текстуры.

Но что делать, если желания учиться моделировать и рисовать текстуры нет, а игры делать очень хочется? В наши дни даже это не означает, что двери в чудесный мир разработчиков игр для вас закрыты: существуют различные онлайн-магазины, именуемые маркетплейсами. Там вы можете купить ассеты – наборы различных уже смоделированных объектов, которые можно использовать в своей игре.

Ассет может включать в себя, например, десятки 3D-объектов на определенную тематику: бар, детская площадка, средневековая деревня. Если автор ассета указывает, что он не против использования этого ассета в коммерческих целях другими людьми, то вместо того чтобы моделировать целую деревню, вы можете ее просто купить на

таких ресурсах, как **Unity Asset Store, Envato, 3dmodelhaven** или **Unreal Marketplace**.

Вы ошиблись, если вы подумали, что есть всего два варианта для создания 3D-игры – или сделать модели самостоятельно, или купить модели за деньги. Третий вариант заключается в том, чтобы пользоваться ассетами бесплатно. На Unreal Marketplace каждый месяц раздается весьма внушительный набор ассетов на самые разные темы и в самых разных стилях. Средняя цена всего раздаваемого набора в любое другое время составляет от 25 до 50 тысяч рублей, а в момент акции вам достаточно просто скачать то, что вам предлагают в этом месяце.

Более того, вы можете просто установить в поиске галочку FREE в разделе «цена» и выбрать что-нибудь из выложенных в общий доступ ассетов. Есть ресурсы, на которых вас вообще круглые сутки ждут бесплатные наборы 3D-объектов: **Oyonale, Turbosquid** и **CGTrader**.

Если вам кажется, что игра, созданная с использованием ассетов, всегда будет высмеяна игроками, то посмотрите на игру **Bright Memory** и почитайте отзывы к ней. Это проект, созданный одним человеком, который ни разу в жизни не занимался моделированием. Все, что вы видите, – это комбинация скачанных и купленных ассетов. Игрокам нет никакого дела до того, кто сделал 3D-модель главной героини или, например, нарисовал горы на заднем плане. Даже крупные студии используют собственные ассеты из проекта в проект. Так, например, в Red Dead Redemption 2 вы сможете найти точно такие же деревья, которые вы видели в GTA V.

Создателем 3D-моделей также на помощь могут прийти быстро развивающиеся нейросети вроде **Instant NeRF** от **Nvidia**, которая способна превращать в 3D-объекты целые улицы из реального мира путем обработки сотен фотографий. **RealityScan** от Epic Games позволит вам с помощью камеры мобильного телефона «сканировать» в 3D любой попавшийся под руку объект – достаточно будет побегать вокруг него, записывая видео или делая фотографии. Разумеется, нейросети сделают работу весьма грязно и неаккуратно, но кому-то подправить детали и заняться ретопом (уменьшение количества полигонов) будет куда проще, чем моделировать все с нуля.

Попробуйте установить себе разные программы и посмотреть ролики об их возможностях, чтобы определиться, какой набор инструментов вам нужен для реализации ваших идей.

7. Наши помощники в 2D

Если вам, как и мне, 3D неинтересно и чуждо, а свой путь к успеху вы планируете проложить плоскими спрайтами вместо трехмерных фигурок, то встанет вполне резонный вопрос: в какой же программе начинать творить?

Самой раскрученной программой для рисования является **Photoshop**, но его цена словно устанавливалась с одной простой целью – отпугнуть новичков. Поиск бесплатного аналога может привести вас к программе **GIMP**, которую, опять же с помощью бесплатных плагинов, можно вполне развить до точно такого же функционала, которым хвастается Photoshop. Помимо GIMP я рекомендую обратить внимание на программу **Krita**, примеры анимации в которой я советую вам посмотреть на YouTube: эти работы у многих вызывают восторженный душевный трепет.

Krita и GIMP прекрасно годятся для создания растровых изображений. Растровое изображение – это файл, представляющий собой мозаику из цветных пикселей (самых маленьких точек, которые может отобразить ваш экран). Откройте фотографию вашего товарища в VK, сделайте свой собственный снимок или нарисуйте и сохраните что-нибудь в Paint – перед вами предстанет «растровое» изображение.

О смену размера картинок обжигаются многие начинающие разработчики, которые не могут определиться с тем, в каком же размере и формате им рисовать объекты для своей игры. Чтобы не обжечься о «мыльные» рисунки, я могу дать весьма точную рекомендацию. Для начала выберите, в каком максимальном разрешении вы хотели бы видеть свою игру. Я бы рекомендовал адаптировать всю графику под fHD – т. е. рисовать все объекты под размер экрана 1920 × 1080 точек. Но тут возникнет проблема, свойственная растру: взгляните, например, на Fearmonium *(рис. 7)*.

Рис. 7. ***Fearmonium, Slava Gris, 2021 год***

Высота моего персонажа – 167 пикселей. Если вы создадите в GIMP или Krita холст высотой 167 пикселей, то в попытке прорисовать мелкие детали вы сможете изобразить только неразборчивую кашу. Несмотря на то что в игре этот персонаж присутствует как объект высотой в 167 пикселей, нарисован он был в размере, куда более высоком.

Я рисую все объекты в размере вдвое больше, чем мне необходимо, и уже только перед самым экспортом уменьшаю размер с выбранными мною настройками интерполяции, т. е. с настройками сглаживания при изменении размера картинки. Я советую вам поэкспериментировать с разными настройками интерполяции в программе, которую вы выберете, и решить «на глаз», какой алгоритм изменения размера изображений вам нравится больше. Определившись с выбором, не отступайте от него – изменяйте размеры ваших рисунков для одного проекта с одними и теми же настройками.

Если же вы хотите погрузиться в атмосферу ретроигр и ваш будущий шедевр планируется быть выполненным в pixel-art, то вышеперечисленных проблем у вас не возникнет: ваш единственно правильный выбор – рисовать объекты таким образом, чтобы квадратный пиксель являлся самым мелким объектом на экране.

Для Catmaze я выбрал разрешение 480 × 270 пикселей. Вся игра рисовалась под этот размер без всяких уменьшений и увеличений: главный персонаж, Алеста, заняла всего 44 пикселя в высоту *(рис. 8)*.

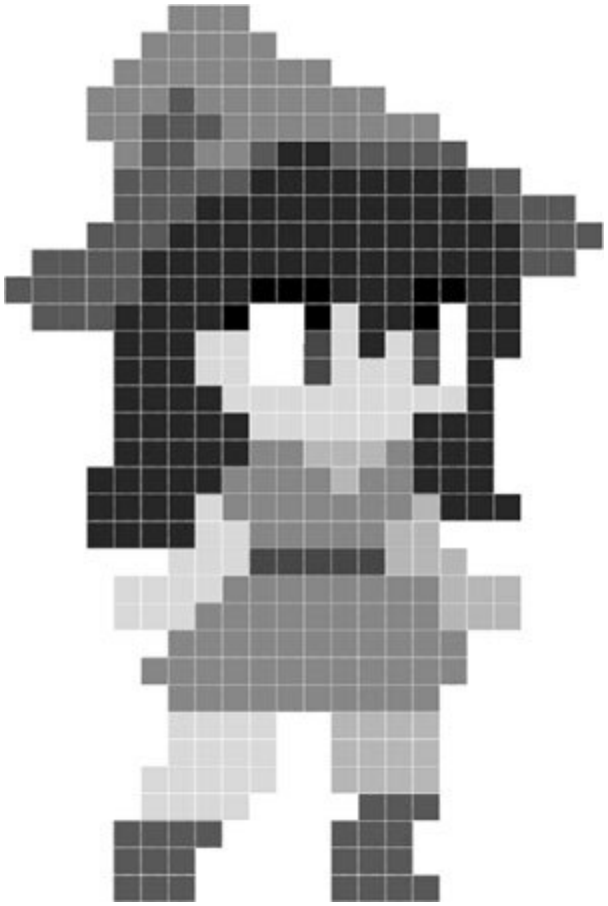

Рис. 8. ***Спрайт Алесты из Catmaze высотой в 44 пикселя***

Для рисования я использовал всего один инструмент – «карандаш» с твердыми краями и размером 1 пиксель. Каждый «квадратик» на этой картинке был кликом мышки. Когда вы видите один экран Catmaze – вы видите приблизительно 129 600 кликов мышкой.

Изображения в пиксель-арте – это все те же растровые изображения, но нарисованные на уровне пикселей, а не появившиеся в процессе размашистой работы кистью с неточными краями.

Несмотря на то что такой инструмент, как «карандаш», есть и во всем известной программе Paint и даже в некоторых движках, позволяющих вам рисовать текстуры, изобретено множество программ, чтобы 129 тысяч кликов мышкой протекли менее утомительно.

Самым мощным помощником в рисовании с помощью пикселей станет **Aseprite**. Программа хоть и платная, но ее цена крайне мала и на момент написания этой книги составляет около 500 рублей. Aseprite будет помогать вам не громоздить лишних пикселей при рисовании линий, а также предоставит кучу инструментов для работы с анимацией, изометрией и интерполяцией. Помимо Aseprite можно присмотреться к **Pyxel Edit** – тоже весьма недорогой программе для работы с пиксель-артом. В отличие от Aseprite, Pyxel Edit больше предназначен для рисования окружения, нежели для работы над персонажами.

Если же вам принципиально, чтобы решение было бесплатным, можно посмотреть в сторону того же GIMP или же **GraphicsGale** – обе эти утилиты предоставят вам необходимый для нашей работы инструментарий.

Но для 2D-графики у нас остается еще и третье решение – забытое, игнорируемое, но не менее интересное, чем растровая графика или пиксель-арт. Я говорю про вектор! Я не предлагаю делать вам Flash-игры, потому что эта технология уже нигде не поддерживается, речь идет об использовании редакторов векторной графики с целью создания растровых изображений. Процесс этот выглядит так: вы рисуете что-либо в векторе, а экспортируете уже в. png (один из форматов для растровых картинок) и в таком виде добавляете в игру.

В векторных редакторах изображение состоит не из пикселей, а из математических формул. Главным плюсом такой структуры является то, что, пока вы не отрендерели изображение (не нажали «сохранить как. png») вы можете манипулировать размером картинки как угодно – на качестве рисунка это не скажется никаким образом *(рис. 9)*.

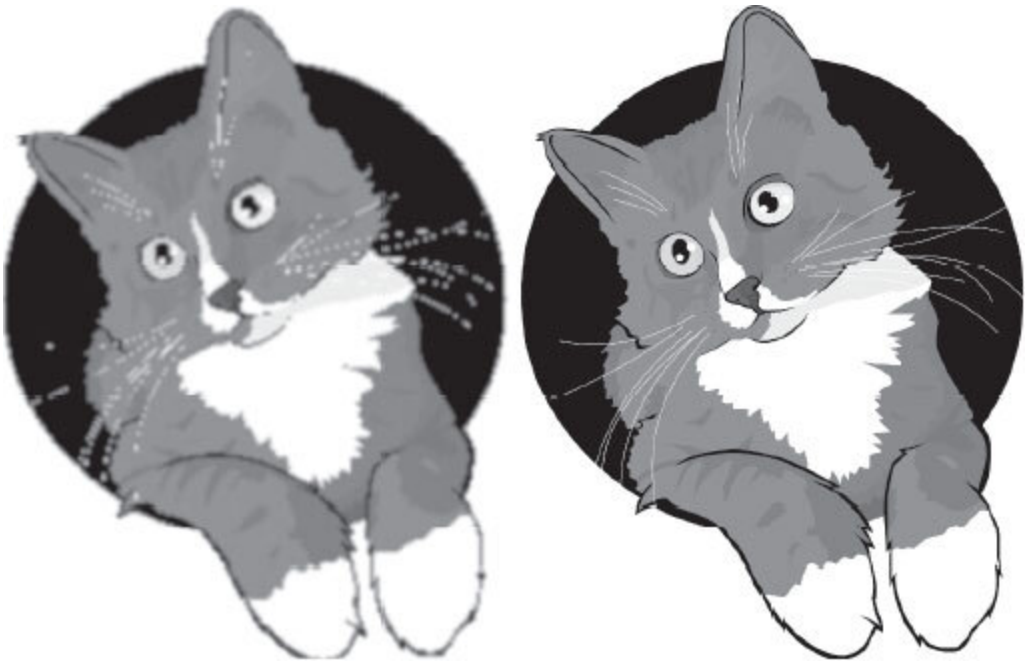

Рис. 9. ***Сильно приближенное растровое изображение vs сильно приближенный рисунок того же размера, но в векторе***

Современное привлекательное платное решение для вектора предоставляет **Adobe Animate**, а бесплатное – программа **Inkscape**. Плюсом работы в векторе является возможность рисовать мышкой и не чувствовать себя сумасшедшим: мы размещаем на холсте прямую линию, а затем с помощью курсора «гнем» ее в выбранных местах. Таким образом, линия всегда остается аккуратной, а не «дрожащей», как штрихи мышкой, выполненные в Photoshop.

Недостатком векторной графики является огромная трудность в создании переходов между цветами или рисовании нечетких контуров: вектор не подразумевает наличия «мыла», тут нет мягких кистей, а использование градиентов усложнено до безумия. Я очень часто довожу свои рисунки «до ума» уже в программах для работы с растровой графикой.

Чтобы не рисовать вовсе, вы можете найти бесплатные 2D-ассеты на просторах Интернета, но их узнаваемость будет куда выше, чем узнаваемость 3D-объектов, чей внешний вид во многом обусловлен постпроцессингом в движке и вашим вмешательством в структуры объектов. Вмешаться в уже нарисованный 2D-объект куда сложнее, а

высокая узнаваемость бесплатных 2D-спрайтов крайне негативно скажется на индивидуальности вашей игры.

Но взять некоторые объекты за основу для своих экспериментов с кистью или найти готовое решение, например, для интерфейса или эффектов, кажется не такой уж плохой идеей. Для покупки 2D-ассетов вам подойдут все уже перечисленные «маркетплейсы»: 2D-ассеты есть как и в Unreal Marketplace, так и в Unity Asset Store и на том же Envato. А вот за бесплатными 2D-ассетами можно заглянуть на **itch.io** (в раздел assets), **kenney** или на **OpengameArt**.

> **Не нужно без подготовки садиться рисовать – на данной стадии нужно лишь определиться с инструментарием и отвести несколько часов на изучение интерфейсов различных программ для рисования, а также ознакомиться с разнообразием бесплатных ассетов.**

Другим способом сэкономить ресурсы на рисовании может стать использование нейросетей. Упоминания достойны три из них: платная **MidJourney**, доступная только по приглашению **Dalle-2**, и непростая в освоении, но зато полностью бесплатная нейросеть **Stable Diffusion**. Все они работают по единому принципу: вы задаете текстовый запрос, который принято называть **prompt**, и на основе ваших слов нейросеть пытается создать изображение. Иной раз работы выглядят очень неплохо, но для того чтобы найти им применение, вам придется подключить всю вашу фантазию, потому что нейросети эти пока что неповоротливы и понимают ваши просьбы или слишком уж буквально, или, напротив, в качестве результата выдают весьма абстрактные рисунки. Нарисовать, например, одного персонажа с нескольких ракурсов или с разными выражениями лица с помощью нейросети у вас на данный момент не выйдет.

Для анимирования можно освоить массивную науку покадровой анимации, которая используется во всех ретроиграх, двухмерных мультфильмах Disney и некоторых независимых проектах вроде всем известного Hollow Knight. Я и сам предпочитаю анимировать изображения по кадру, несмотря на то что это занимает огромное количество времени, но если вы спешите доделать свою первую игру

(что является абсолютно правильным решением), я бы рекомендовал присмотреться к «скелетной анимации».

«Скелетная анимация» представляет собой процесс, для которого вам необходим персонаж, состоящий из мелких частей. Ноги, стопы, голова, волосы, глаза, руки, предплечья и т. д. должны быть нарисованы отдельно. Все эти кусочки собираются в скелет, и вам дается возможность манипулировать ими отдельно. Для того чтобы персонаж, например, поднял руку, вам не нужно рисовать кучу кадров анимации, а достаточно просто потянуть его за кисть. Примеры такой анимации вы можете посмотреть в играх студии Vanillaware – 13 Sentinels: Aegis Rim, Odin Sphere и Dragon's Crown. Из платных приложений, позволяющих работать со скелетной анимацией, лидирующей по функционалу является программа **Spine**, а из бесплатных – **Dragon Bones** и, как ни странно, игровой движок Unity, предоставляющий достаточно массивный набор инструментов для работы со скелетной анимацией прямо внутри движка.

Вам самим решать, к чему больше предрасположен ваш склад ума и с каким типом графики вам будет проще работать: если, как и у меня, ваше мышление в 3D оставляет желать лучшего, а инженерная мысль не блещет искрами гениальности, то выбирать плоский холст для выражения своих идей – не худший вариант. Множество отличных современных игр делается в 2D.

Как научиться рисовать, моделировать и работать на движках – мы еще разберемся, но какой бы способ выражения своих мыслей вы ни выбрали, нас всех, и 3D-, и 2D-специалистов, объединит работа со звуком.

8.

8. Я не слышу, дети…

В классическом процессе создания музыки замешано множество людей: и музыканты, и композиторы, и звукорежиссеры. Создание музыки представляет собой сферу, не менее сложную, интересную и глубокую, нежели создание игр.

Нельзя недооценивать влияние музыки на вашу игру. Качество звука в игровых проектах во многом формирует атмосферу и общее настроение. Музыка может даже определить то, чем будет заниматься ваш игрок: например, если в игре внезапно спокойная мелодия сменится эпичным военным маршем, то игрок начнет суетиться и искать угрозу, а когда стихнет последняя звонкая скрипка и зазвучит умиротворенная сельская мелодия, игрок поймет, что битва закончена.

Есть несколько вариантов по решению проблемы с музыкой. Можно создать ее самому, как это сделал автор Cave Story, автор Iconoclasts и автор Undertale. Мы с вами легких путей не ищем, потому я остановлюсь на этом варианте подробнее – лично я написал несколько композиций самостоятельно и попробовал парочку программ.

Самой популярной и простой в освоении является, конечно же, **FL Studio**, известная «старичкам» как **Fruity Loops**. Но так же, как и от Photoshop, FL Studio отпугивает сумасшедшей ценой на лицензию. Наш взор придется сначала обратить на бесплатные аналоги.

Первым достойным упоминания программным обеспечением для написания музыки является **LMMS**. Очень многое в нем схоже с FL Studio, так что освоение этой бесплатной программы поможет вам в будущем разобраться и с более профессиональными приложениями. **SunVox** предоставит немного более ограниченные возможности, но, когда вам нужна музыка лишь для заднего плана, местные ограничения только сыграют вам на руку, вынуждая останавливаться на весьма незамысловатом звучании.

Если вы делаете игру в ретростиле, то можно пойти очень интересным и неординарным путем: установить себе эмулятор портативной приставки GameBoy и запустить на нем **Little Sound DJ** – так вы получите аутентичный 8-битный звук из ушедших эпох, а

заодно познакомитесь со всеми ограничениями, в которых приходилось работать в самом конце восьмидесятых.

Неплохими приложениями являются **FamiStudio** и **Bosca Ceoil**, которые позволят вам поработать с «чиптюн»-музыкой и имитировать звучание старых консолей, но уже используя современные методы.

> **Посмотрите на YouTube уроки обращения с каждой из перечисленных программ и определитесь с тем, в какой из них звучание наиболее соответствует вашей задумке.**

Для того чтобы написать композицию, играющую в главном меню Catmaze, мне потребовались знание о том, что такое «минор» и «мажор», две недели и добрые советы моего хорошего друга-музыканта Expecte Amour, на которого я свалил обязанности по написанию всех остальных композиций к своим играм.

В решении делегировать написание музыки другому человеку и заключается второй подход к обеспечению вашей игры звуковым наполнением: вам не обязательно использовать музыку, которую создали вы сами. У нас есть несколько способов обеспечить свою игру музыкой, не притрагиваясь к нотной грамоте.

Первый способ заключается в том, чтобы на ресурсах вроде того же opengameart найти бесплатную музыку. Обращайте внимание на лицензию, под которой распространяется композиция, и обязательно следуйте всем требованиям по указанию авторства. Я рекомендую присмотреться к лицензии «cc-by 4.0», разрешающей коммерческое использование композиции при условии, что где-либо в вашем проекте будет упомянут автор, а также редкой удачей будет найти что-то подходящее под лицензией «CC-0», предоставляющей вам безграничные возможности по использованию и изменению композиции.

Второй способ: купите нужную музыку на сайтах вроде уже упомянутого Envato. С одной стороны, выбор платных композиций значительно шире, чем бесплатных, а значит, и вероятность подобрать что-то более подходящее вашему продукту куда выше. Но, с другой стороны, даже с наличием лицензии на использование находящейся в общем доступе композиции вы можете попасть в очень глупую ситуацию: композиция продается на стоках, а значит, что ее приобрели

не только вы. Если другой покупатель включит ее в саундтрек своей игры и решит опубликовать этот саундтрек отдельно на каком-нибудь сервисе вроде Spotify, то ему придется обращаться к специальным дистрибьюторам, которые обеспечивают появление музыки на подобных интернет-площадках. Для распространения композиции и борьбы с конкурентами дистрибьютор закрепит права на трек за собой, и случится страшное: всем, кто использовал эту композицию в своих видео на YouTube или Twitch, придет оповещение, что с их ролика с этой музыкой теперь исчезает монетизация. Чудовищность такого исхода событий заключается в том, что YouTube и Twitch – это очень массивные площадки для раскрутки вашей игры. Если блогеры при попытке загрузить видео с обзором или прохождением вашего проекта будут лишаться монетизации, то мотивации распространять информацию об игре у них не будет никакой.

Лучше подумайте о том, чтобы прибегнуть к самому интересному способу придать вашей игре уникальное звучание: можно найти непопулярных музыкантов на просторах Сети и предложить им сотрудничество. Подавляющему большинству музыкантов будет даже приятно предоставить вам свои композиции или бесплатно, или за символическую сумму. Для них участие в вашем проекте может стать символом признания, а выход игры в свет вполне может открыть им новые двери и предложения к уже коммерчески выгодному сотрудничеству.

Мое лихое описание того, как все на этом свете просто и что в наши дни студию из каскадеров с дорогим оборудованием заменит Cascadeur, штат из десятка аниматоров уместится в ярлык Spriter Pro, а целый оркестр поместится в LMMS, вполне может обидеть тех, кто работает с одной из перечисленных мною программ и твердо убежден, что его дело – сложное и уважаемое. И в этой обиде кроется кое-что очень плохое и неправильное…

9. Разрушительное самолюбие

Что же в итоге отличает нас от Сакагучи и Миямото, задавших направление развития ключевых игровых жанров в конце 80-х? Почему, имея в распоряжении такие удобные программы и вычислительные мощности, современные студии состоят из сотен людей?

Можно было бы сказать, что Final Fantasy и Super Mario – технически куда более простые игры, нежели то, что мы разрабатываем в наши высокотехнологичные дни, и мое сравнение некорректно. Только стоит иметь в виду, что технологии, которые сейчас кажутся простыми и примитивными, в конце восьмидесятых являлись таким же чудом, коим в наши дни является, например, Ray Tracing (система реалистичного отражения лучей). Через 30 лет вещи, которые в данный момент вызывают у нас трудности, будут казаться такой же мелкой и незначительной проблемой, как, к примеру, размещение на карте более 25 % уникальных спрайтов.

Те, кто не согласен, что в настоящее время разработка игр протекает чуть ли не играючи и не требует высоколобости, являются одной из серьезных проблем для успешного создания игр.

Я как-то переводил статью моего товарища и выкладывал ее на англоязычный Reddit. Он был автором весьма успешного ужастика от первого лица, разработанного на Unreal Engine. Игра называлась Never Again, а в статье описывался его тернистый путь «от идеи до выпуска».

Объясняя, почему он решил работать именно на Unreal Engine, мой знакомый разработчик использовал фразу «я не программист, никогда на программиста не учился, потому мой выбор пал на Unreal Engine с его системой Blueprints». И это хорошая фраза, емкое объяснение и отличная мотивация: я не уверен, что каждый из читателей этой статьи являлся программистом, но в том, что многие хотели бы сделать игру того же уровня и добиться того же успеха, – сомневаться не приходилось. Я вижу в этой фразе луч надежды для тех, кто очень хочет делать игры, но абсолютно не знаком с программированием.

После публикации статьи стали появляться первые комментарии. Один из комментариев столкнул меня с самым ужасным, деморализующим, высокомерным и претенциозным мнением, которое максимально разрушительно и вопиюще абсурдно. Автор этого комментария написал: «So I ask you to withhold from discrediting people using blueprints», что дословно можно перевести как: «Я попрошу вас воздержаться от дискриминации людей, использующих Blueprints».

В более развернутом объяснении своей позиции автор рассуждал о том, какими высокими знаниями, навыками и опытом нужно обладать, чтобы уметь перетягивать мышкой квадратные окошки. Я не буду спорить, что знания, сообразительность и опыт для освоения визуального программирования все-таки нужны. Просто количество требований к навыкам и умениям для создания игр сейчас смехотворно мало по сравнению с тем, что требовалось от разработчиков раньше.

Считать «ущемлением» своих навыков высказывание, что работа с Blue Prints подойдет не только программистам, – нелепо и разрушительно. Такая позиция вредна.

Чем выше мы оцениваем свои достижения, тем сильнее мы довольны собой и тем сильнее нам хочется себя наградить. Чаще всего наградой выступает безделье или хотя бы приостановка получения новых знаний, потому что «я ведь и так сделал достаточно». Горделивое самодовольство – это не то, что может послужить двигателем как в работе, так и в познании. У вас будет очень комичный вид, если вы освоите простой навык и после этого мысленно поставите себя на один уровень с теми, кто потратил несколько лет на изучение чего-то более низкоуровневого, чем визуальное программирование. С таким подходом возможность стать настоящим специалистом сведется на нет. Вы так и останетесь парнем с Reddit, который будет обижаться на фразы вроде «blueprints – это не программирование».

Людям, которые далеки от разработки видеоигр, достаточно бегло глянуть на Fearmonium и услышать, что я в этой игре не отвечал только за создание музыки, как они видят во мне многорукого волшебника, обладающего вагоном знаний и опыта. Безусловно, приятно, когда тебя воспринимают высококвалифицированным специалистом. Но важно понимать, что в нашей ситуации это – ложь.

Те, кто хотя бы чуть-чуть погружен в разработку видеоигр, прекрасно понимают, какой ерундой я на самом деле занимаюсь: я вожусь с детским движком Construct и сгибаю векторные линии мышкой, когда рисую. Это и рядом не стояло с тем титаническим трудом, которым занимались наши «предки» в лице гениального Сакагучи и Миямото. У меня нет и одной сотой знаний и навыков, что были у них.

Но, безусловно, поощрять себя нужно. Я не предлагаю отказываться от гордости за свои успехи. Освоить Blueprints – это все-таки далеко не пятиминутное дело. Есть условия, при которых гордость и высокое мнение о собственных навыках не наносят вреда. В перечень таких условий нужно обязательно включить понимание предела своих возможностей. Когда перед нами встает новая задача, мы оцениваем ее сложность, ориентируясь на свое прошлое. Поверьте, когда вы только учились ходить, принять решение «дойти до мамы» давалось с трудом, потому что ваш опыт ходьбы заключался в бесконечных падениях, шишках и фрустрации. Куда проще доползти до мамы на четвереньках. Но ходить вы все-таки научились, и теперь это действие не представляет проблемы. Ваш опыт подсказывает вам, что ходить не так уж и трудно. Теперь это даже проще, чем ползти!

В пример можно привести и катание на велосипеде, и игру в видеоигры: любое действие, которое уже завершалось успехом хотя бы один раз, дается нам куда легче в следующий. К нам приходит осознание собственных возможностей, мы обретаем нужный опыт и уверенность, что у нас все получится и в следующий раз.

Представьте, что вы уже изучаете движок и проходите курс, разбитый на несколько равных по сложности видеоуроков. Если в течение первого дня обучения уже завершен аж один видеоурок, то появляется объективная причина порадоваться и передохнуть. Но в случае если вы уже давно занимаетесь прохождением курса и в вашей жизни значится день, когда успешно было пройдено 19 видеоуроков, то такое достижение, как «я завершил один видеоурок», перестает восприниматься поводом для гордости: у вас уже есть осознание, что можно делать больше. Вы познали свой лимит.

Достижения из прошлого чертовски сильно влияют на то, каким вам видится горизонт возможного в настоящем.

Я встречался с маркетологами, которые тратят весь день на написание двух-трех маркетинговых писем. Они твердо убеждены, что выполняют трудную работу и делают действительно много. Именно **преждевременное чувство удовлетворения** от своей деятельности мешает им делать больше. Я отправлял по 30 маркетинговых писем за день, каждое из которых не было точной копией предыдущего. И я даже не устал. Потому что за день до этого я отправил 40 таких писем.

На то, какими перед нами предстают наши возможности, влияют и достижения других людей, которые мы очень внимательно и вдумчиво проецируем на себя. Спортсмен по имени Томас Берк в 1896 году пробежал 100-метровку за 12 секунд, установив мировой рекорд. За прошедшее время люди не отрастили себе дополнительных ног, никаким образом не мутировали, не начали бегать на руках, но в наши дни пробежать 100 метров за 12 секунд уже не означает взять золото на Олимпийских играх. Это теперь норматив для всего лишь третьего спортивного разряда. И знающие люди могут бросить в меня аргументом про ручные и автоматические секундомеры, но погрешность в вычислениях не была столь уж значительна, чтобы оспорить мою мысль – Томас Берк продемонстрировал лимит в 12 секунд. Позже это значение спустилось до 9 секунд, и тогда прежний результат перестал выглядеть как суровый, невыполнимый вызов. Задача, растеряв свою грозность и ореол невыполнимости, становится просто проходной рутиной.

Человек, знающий, на что он способен, делает намного больше.

Между пределами, которые поставил перед собой парень с Reddit и лимитами, которые установил себе Сакагучи, – неимоверно глубокая пропасть. И мне хотелось бы грубо и грозно отрезать, что пропасть эта состоит из жалости к себе, слабой воли, отягощающей лени и вопиющего лицемерия, но все не так однозначно. Причины, по которым мы работаем мало, плохо и медленно, лежат куда глубже.

Многие боятся заходить очень далеко в проверке своих возможностей из-за страха «сгореть». На меня такие люди смотрят снисходительно и жалеют за то, как я несчастлив в своей работе и как близок к выгоранию. Но, во-первых, я счастлив, а во-вторых, причины выгорания кроются отнюдь не в работе. Выгорание наступает не потому, что мы много вкалываем, а потому, что мы много волнуемся и неправильно отдыхаем.

Мы еще поговорим о выгорании, но, перед тем как научиться не сгорать, нужно научиться работать.

Сосредоточьтесь на мысли, что разработка видеоигр – это увлекательный и интересный процесс, донельзя упрощенный современными технологиями.

10. Учиться, учиться, учиться

Обучение – это путь к силе.

Tales of Symphonia, Кратос

Перед тем как познавать пределы своих возможностей, необходимо набраться хотя бы минимальных знаний и навыков. Мы не сможем пробежать и сотни метров, если не умеем ходить. Но вот беда: сейчас рядом с нами нет добрых взрослых, которые возьмут нас за крошечные ручки и помогут удержать равновесие. Здесь только мы и необъятных размеров океан разрозненных знаний про разработку видеоигр. Пересечь этот океан можно тремя путями.

Первый путь: сейчас знания и навыки можно приобрести в государственных учреждениях как по программе бакалавриата – для тех, кто только окончил школу, так и по программе магистратуры – для тех, кто уже может похвастаться наличием диплома о высшем образовании. Не такое уж и большое количество вузов идет в ногу со временем столь точно, чтобы внедрить программы по обучению разработчиков игр, но тем не менее выбор у абитуриента есть.

В Москве можно присмотреться к Высшей школе экономики, где можно поступить на направление «Гейм-дизайн и виртуальная реальность. Системный гейм-дизайн». Другим вариантом может послужить Московский политехнический университет с направлением «Программное обеспечение игровой компьютерной индустрии». В Санкт-Петербурге можно поступить в Университет промышленных технологий и дизайна на специальность «Анимация и графика компьютерных игр» или же в ИТМО на «Технологию разработки компьютерных игр». Порадовать наличием прогрессивных специальностей могут и Институт дизайна и сервиса в Челябинске, и даже Вятский государственный университет в Кирове.

Наиболее актуальную информацию, конечно, стоит искать уже к моменту поступления, а не надеяться, что в одном абзаце я перечислил все вузы, где есть столь современные кафедры. Но у государственного

образования имеется ряд громоздких изъянов, о которых я не могу умолчать.

Управленческий аппарат в больших учреждениях – это весьма неповоротливая структура, в то время как игровая индустрия чрезвычайно изменчива.

Вот вас учили решать проблемы при работе с Unreal Engine 4, а в день вручения диплома компания Epic Games выпускает Unreal Engine 5 – новую версию движка, где старые проблемы решены, а новые пока не изучены.

Директора множества игровых компаний хоть и неплохо относятся к наличию у сотрудника диплома о высшем образовании по направлению «разработка игр», но и не считают такой атрибут обязательным для приема на работу. Для них важна ваша способность решить конкретную тестовую задачу, и в слове **«конкретный»** и кроется подводный камень высшего образования: создание игр – это отнюдь не тот процесс, который можно унифицировать.

Каждый игровой проект – индивидуален, а каждый разработчик сталкивается с уникальным и неповторимым набором проблем. Столь досадная аксиома обесценивает опыт преподавателей, которые физически не смогут предусмотреть необъятное многообразие трудностей, с которыми столкнутся их студенты при разработке собственных игр.

Плюс ко всему «уметь» и «учить» – это совсем разные, даже независимые друг от друга навыки. В государственных учреждениях нет такого направления, как «педагог по игровому дизайну». Это, на мой взгляд, противоречит существованию направлений, обучающих разработке игр. Кадры, которые умеют и делать игры, и **учить** делать игры, являются чрезвычайно редким ресурсом.

Второй путь получения знаний – это дополнительное образование, представленное в России в виде множества платных курсов. Такой курс не будет длиться дольше, чем несколько месяцев, а программу курса чаще всего разрабатывает тот же человек, который этот курс и ведет. Такой подход позволяет максимально быстро подстраиваться под изменчивую реальность и не тратить время на изучение устаревшей среды разработки.

Курсы бывают весьма узконаправленными. Вам не придется тратить время на что-то общеобразовательное. Можно сразу обозначить те

вещи, которые вам необходимо знать для успешного завершения вашего личного проекта. Берете курс «Дизайн 2D-персонажей», прибавляете к нему курс «Написание игровых сценариев», и у вас появится достаточно навыков, чтобы **начать** создавать свою визуальную новеллу.

Возможность пройти образовательные курсы звучит очень соблазнительно. Курс снимает с вас ответственность за собственное обучение – добрый и умный учитель проведет вас за ручку по изнурительной тропе образования, а ваши однокурсники будут поддерживать друг друга советом и веселой беседой.

Но все не так радужно. Я сам вел курсы, и сам задавал себе страшный вопрос, которым потом пугал других знакомых мне преподавателей: а много ли ваших студентов продолжили делать игры после курса? Преподаватели не любят отвечать на подобное, некоторые даже воспринимают ушедших из индустрии студентов как личную неудачу. И, честно скажу, «неудач» у каждого из нас очень много: после прохождения курсов по разработке игр в индустрии остается меньше 10 % слушателей.

Проблема заключается не в квалификации преподавателя, и даже не в самом студенте. Тайна исчезнувших из индустрии кадров куда глубже, интереснее и сложнее.

Давайте вспомним о слове «конкретный». Именно уникальность и неповторимость задач, которые возникнут перед вами в ходе разработки собственной игры, станут решающим фактором в пополнении очень тоскливой статистики. Никто и никогда не сможет вам объяснить, как сделать конкретно ту игру, которую хотите сделать именно вы.

Даже если вы окончите вуз и пройдете курсы, то в любом случае наступит момент, когда вы зайдете достаточно далеко в разработке игры и с досадой осозна́ете: к решению незаурядной и индивидуальной проблемы вас никто не готовил. Как говорится, «это мы не проходили, это нам не задавали…»

И тогда, в поисках решения непредвиденной проблемы, вам придется прибегнуть к навыку, приобретение которого является ключевым для разработчика одиночки и представляет собой наш третий путь. Навык этот носит грозное и гордое имя **самообразование**.

В вузах и на курсах вас могут научить делать 3D-модели, писать сценарии, анимировать персонажей, но все это меркнет перед умением получать знания самостоятельно. Тот, кто **умеет учиться**, не нуждается ни в преподавателях, ни в государственных учреждениях. Он не будет спотыкаться о проблемы, решение которых не разбиралось на занятиях. Он будет учиться их решать. Разумеется, в данном случае я говорю исключительно о разработке игр. (Если вы хотите стать, например, врачом или пилотом, то настоятельно рекомендую все же получить высшее образование. Я не хочу умереть в вашу смену.)

Если визуализировать обучение как необходимость перепрыгнуть глубокую пропасть, то все, что сделает преподаватель, – это позволит вам постоять на краю подольше. И пока вы мнетесь перед неизведанной бездной, вы лишь откладываете необходимость приобрести тот единственный навык, который поможет вам освоить почти все на свете, – умение получать знания самостоятельно.

Казалось бы, что учиться тут и нечему: в нашем распоряжении множество ресурсов! Вы знаете, сколько книг написано про разработку игр? Книги научат вас игровому дизайну, построению уровней и маркетингу. Авторы будут стремиться провести вас по извилистой тропе к желаемому успеху – сиди да впитывай знания.

Того, сколько уроков и курсов существует на YouTube, – не счесть. Просто вбейте в строке поиска «уроки Unity», «уроки Unreal Engine», «уроки Construct».

Если заглянуть на официальный сайт выбранного движка, всегда можно найти там исчерпывающую документацию, а иной раз и уже составленный последовательный курс.

Можно отыскать решение многих необычных проблем на просторах Сети, а если утомил поиск, то всегда есть возможность заглянуть на форум разработчиков и переложить ответственность за решение возникшей трудности на плечи более опытных товарищей: Gamedev.ru, c2Community.ru, Prodevs.ru и т. д. А уж количество групп в ВК переходит все мыслимые границы. Используйте для поиска или названия движков, или фразы вроде Gamedev, Gamedevelopment, «Разработка игр» – вам откроются многолюдные сообщества таких же увлеченных ребят, как и вы. Бесплатные лекции; бесплатные циклы видеоуроков; бесплатные материалы, предоставляемые самими разработчиками движка на безвозмездной основе; тысячи форумов с

ответами на десятки тысяч вопросов – все это в нескольких кликах мышкой! Так в чем же проблема? Зачем нам курсы и вузы? Зачем мне писать столько страниц текста про то, как совладать со столь массивным объемом информации?

А потому что совладать с ним действительно сложно: без внимательности и дисциплины уроки и сообщества лишь пополнят список ваших закладок в браузере.

Покупая курсы, вы на самом деле не приобретаете знания, потому что знания – они повсюду. Разработчики движков не прячут способы работы с их программами куда-то в таинственные дебри, доступные только посвященной касте преподавателей курсов. Все лежит в свободном доступе. Талантливые художники и разработчики уже раскрыли свои секреты – они ведут стримы, выкладывают обучающие видео, делятся знаниями.

Вместе с приобретением курсов покупается куда более важная вещь – **дисциплина**. Вы платите не за знания, а за кнут, которым вас к этим знаниям будут подгонять. «Кнутом» я называю неизбежно следующее за вашим бездельем наказание, которое дает внешний стимул заниматься и учиться. Кнутом может стать негодование преподавателя; стыд перед коллегами; осознание того, что куча денег была потрачена впустую, или любая другая причина, которая будет мотивировать вас досидеть на этом курсе до последнего занятия.

Без «кнута», который держит в руках кто-то другой, на пути к знаниям необходимо постоянно самостоятельно держать себя в тонусе. Получив свои корочки и дипломы, вчерашние студенты ужасаются глубине и хаотичности информационного океана, где они остались без поддержки мудрых учителей. Угроза получить за безделье «кнутом» исчезла. Куда и зачем дальше плыть – непонятно.

> **Поступая в университет или покупая курс, вы не только обеспечиваете себе доступ к знаниям. Знания – бесплатны и они повсюду, вам нужен лишь доступ в Интернет. Вы приобретаете себе надзирателя и дисциплину.**

11. Тяжело в учении

Я не буду его ничему учить! Он достаточно умен, чтобы самостоятельно разобраться, как пользоваться книгами.

Record of Lodoss War: Advent of Cardice, кузнец Анвар

Заглянув хотя бы одним глазком за кулисы профессии разработчика видеоигр, можно ужаснуться тому, как много нужно уметь, даже с учетом того, что бóльшую часть работы за нас сделают программы, дел все равно остается невпроворот: игровой дизайн, маркетинг, локализация, сотни анимаций, тысячи 3D-моделей… Знание об объеме предстоящих трудностей – губительно.

Представьте, что вы проголодались и вспомнили, что на кухне в вашей квартире лежит вкусная шоколадка. Ваш мозг знает, что, для того чтобы добраться до соблазнительной сладости, вам нужно всего лишь встать, пройти пару метров, и вот вы уже шуршите сверкающей оберткой.

Представляя такую примитивную программу действий: вы встаете, открываете дверь на кухню, а там… шоколадку охраняет могучее трехглавое чудовище. Вступив с ним в неравный бой, вы проваливаетесь под пол. Оттуда – в темное и смрадное подземелье. Одолев монстра в свирепом бою, вы тратите три дня и три ночи на поиск выхода из подземелья и наконец-то добираетесь до шоколадки.

Принял бы ваш мозг решение «сходить за шоколадкой» столь же легко и безоговорочно, если бы знал все подробности о тех суровых испытаниях, что его ждут? Едва ли. Ваш мозг предпочтет сберечь свои ресурсы и минимизировать вероятность быть истощенным в подземелье до голодного обморока.

Этим примером я хотел рассказать немного о том, как мы принимаем решения, и я не для красного словца представил вас и ваш мозг разными персонажами. Дело в том, что время, когда вам

захочется «съесть шоколадку» или «начать изучать визуальное программирование», по изначально неведомому нам наитию определяет именно мозг. Мы лишь покорно исполняем его волю.

Нейробиологи давно обнаружили, что решение человека становится осознанным только через 10 секунд после того, как мозг уже твердо его принял. Доказывающий эту теорию эксперимент заключался в том, что испытуемым предоставили возможность нажимать на одну из двух кнопок (под правой и левой рукой), а параллельно – показывали им разные буквы на экране с интервалом в полсекунды. В момент, когда человек – участник эксперимента – решал нажать на кнопку, он должен был запомнить букву с экрана и позже назвать ее ученым. Во время всей этой странной вакханалии ученые замеряли активность в коре головного мозга испытуемых. Намерения двигать правой или левой рукой соответствуют различным сигналам в мозге, что позволяло ученым сопоставить, когда решение нажать на кнопку было реально принято мозгом, а когда – осознано участником эксперимента.

«Разрыв в осознании» составил около 10 секунд. Решения принимает наш мозг, а через 10 секунд мы гордо выдаем это решение за свое собственное. Это напоминает ситуацию, когда на многолюдном и шумном застолье кто-то очень удачно пошутил, но шутку расслышал лишь сидящий рядом товарищ и через пару секунд выдал вашу остроту за свою, собрав овации и громкий смех.

Решение сесть учиться, начать работать или залипнуть в социальной сети наш мозг принимает, не советуясь с нами. В понимании того, как он работает, и кроется возможность преодолеть большинство наших проблем. Потому первое, что стоит усвоить на пути к выпущенной игре, – это принципы работы своего мозга.

Согласитесь, человек, который занимается сваркой, должен иметь представление о том, как работает сварочный аппарат; человек, который работает таксистом, должен понимать, как управлять автомашиной.

Разработка игр – это интеллектуальный труд. Наш сварочный аппарат (или машина) – это наш мозг. Но ни на каких курсах по разработке игр работать головой вас не научат. Без понимания того, откуда берется мотивация и почему она может улетучиться, вероятность три года просидеть за компьютером, создавая свою игру и довести ее при этом до конца, уменьшается в десятки раз.

И работа, и обучение без знания дела превращается в хаос: иной раз учиться очень хочется, и этот процесс затягивает; в другой раз, вместо того чтобы потратить свободные часы на образование, мы залипаем на YouTube, а в третий раз – работаем на одной только силе воли, изматывая себя и не получая от процесса никакого удовольствия.

У многих знакомых мне разработчиков работа течет по наитию и привязана к их настроению. Из-за этого ребята работают крайне неспешно, а когда приходит время в срочном порядке исправлять ошибки в игре или торопиться с ее выпуском – им приходится мучиться и заставлять себя вкалывать через силу под давлением паники и стресса.

Я не буду вдаваться в очень глубокие подробности и пробегусь сейчас только по тому срезу знаний, который обязательно пригодится для понимания самого себя.

В нашей голове есть клетки, которые называются нейронами. Они получают и принимают сигналы, а также формируют собой скопления, которые можно назвать «центрами принятия решений». Для краткости обзовем их «ядрами».

«Ядра» были очень здорово визуализированы студией Pixar в мультфильме «Головоломка». Разные колоритные персонажи определяли манеру поведения и ход мыслей человека, в голове которого они заседали. Ядра и правда себя так ведут. Каждое из них отвечает за удовлетворение определенных потребностей: в общении, сексе, статусе, безопасности и т. д.

Если вы решили делать игру, то это решение было принято в «ядрах», а мыслительный аппарат остался в стороне. Ядра решают, читать вам эту книгу дальше или нет. Если во время обучения захотелось отвлечься на смартфон – то, опять же, «спасибо» ядрам.

Все, что человек себе придумывает, отправляется на обсуждение в центры принятия решений. Вы, как я понимаю, решили разрабатывать игры. То ядро, которое отвечает за ваше положение в иерархической структуре общества, начнет поддерживать эту идею аргументами в духе «разработка игр поднимает наш статус!», «выпущенный проект принесет нам денег!», а ядро, которое отвечает за безопасность, тихонечко, но при этом не менее убедительно парирует ему фразой: «А вдруг денег игра не принесет? Вдруг ничего не получится, и мы просто потеряем силы и время, и ресурсов на выживание у нас совсем не

останется?» К нему присоединяется ядро, отвечающее за удовольствие, и предлагает нам открыть ленту новостей в какой-нибудь социальной сети, чтобы посмеяться над старыми картинками.

И вот вместо работы или обучения бездумно листается лента в социальной сети. Мы даже не успели понять, какой эмоциональный и пылкий спор развернулся внутри нашего черепа. Нам показалось, что залипнуть в телефон было самостоятельно принятым решением.

Ничего мы не решали. Это всё ядра, диалога которых даже не слышно, – все произошло очень быстро, неосознанно и без нашего участия.

Из всей этой информации можно сделать вывод, что человек абсолютно не контролирует своего поведения, что, разумеется, ошибочно. Мы не контролируем свое поведение напрямую, но можем определенным способом влиять на центры принятия решений и подталкивать ядра к тому, чтобы их спор всегда заканчивался положительным для нас результатом.

Один из фокусов заключается в том, что аргументы, которые ядра используют друг против друга, сформированы не на почве их собственных умозаключений, а на наших с вами **убеждениях**. Ядра не генерируют информацию; они не придумывают идеи и не делают никаких выводов самостоятельно. Они используют лишь те знания, которые уже есть в нашей черепной коробке.

Поведение контролируется убеждениями. И до тех пор, пока у нас в голове присутствует мысль, что самообразование – это океан, который не переплыть в одиночку, желание сесть и делать игры будет ускользать, а усидчивость во время просмотра обучающих роликов сведется на нет.

Странность состоит в том, что человечество на самом деле всегда знало о такой особенности своего мышления.

Воспитание человека всегда заключалось в прививании ему одобряемых социумом убеждений. Например, ребенку объясняют, что воровать нельзя не только потому, что за этим поступит наказание, но и потому, что это аморально. Если такое убеждение закрепится достаточно прочно, то ядра этого человека при обсуждении насущных проблем никогда не прибегнут к аргументу «зачем нам работать вообще? давай-ка просто наворуем!» Человек даже не услышит призывов к воровству, он просто о нем не подумает.

Люди использовали этот трюк с убеждениями, чтобы человеческое общество не погрязло в хаосе. Благодаря воспитанию мы даже не задумываемся на полном серьезе о том, чтобы действительно совершить что-то аморальное и жестокое. Если безумные идеи и возникают, то ядра, вооруженные убеждениями, что так делать нельзя, быстро растопчут все гадкие и аморальные побуждения.

Но почему мы думаем, что процесс воспитания уже закончился? Если нашим воспитанием больше не занимаются учителя и родители, это означает лишь то, что теперь им занимаемся мы сами. Мы должны закладывать полезные нам убеждения в свою голову самостоятельно. И так же самостоятельно вычеркивать убеждения вредные.

Убеждение, что поглотить огромное количество знаний мучительно сложно, присутствует у многих.

Во время своего обучения я находился в ситуации, которую, как ни странно, можно назвать куда более выгодной, чем современный порядок вещей. Если сейчас у вас есть книги, курсы, образование в вузах, уроки на YouTube и форумы, то, когда я сел изучать Construct Classic более десяти лет назад, все, чем я располагал, – это официальная документация на очень сложном английском. Документация эта не представляла собой последовательной и структурированной базы знаний. Передо мной был скорее перечень функций и возможностей движка в pdf-файле листов эдак на пятнадцать.

Пугал ли меня объем необходимых для усвоения знаний? Отнюдь. Пугало ли меня то, насколько сложно делать игры с таким крошечным количеством информации? Нет. Я тогда понятия не имел, как делаются игры и как мне будет сложно разобраться, что к чему. Сел бы я возиться с прыжком, если бы знал, что впереди меня ждет неделя зависающих анимаций, плохой физики, истерик и фрустрации? Думаю, что нет – я бы не пошел за шоколадкой, если бы знал, что меня ждет трехглавый монстр.

Ситуация, в которой вы находитесь, информация, которая вас окружает, и знания, которые у вас уже есть, – оказывают мощное влияние на формирование новых убеждений. Если не создавать вокруг себя информационного поля, которое способствует появлению твердой убежденности, что вы можете научиться всему самостоятельно, то и

нужные убеждения в голове не отложатся. Не меняя ничего вокруг себя, невозможно поменяться самостоятельно.

В случае когда информация о свирепом чудовище уже просочилась в ваш ум, чертовски сильно помогает концентрация на «настоящем». Нет смысла пытаться выпить весь океан за раз или держать в голове, что рано или поздно вам предстоит это сделать. Решая задачу, думайте только и только о ней, никогда не стоит спешить и забегать вперед. Любая глобальная цель достигается решением сотни маленьких, конкретных вопросов. Вы решили сделать платформер? Погрузитесь сначала в мелкую «лужицу» информации о прыжках, потом в такую же «лужицу» информации о скрытых механиках передвижения, а затем потихоньку осваивайте дизайн уровней. Вы и не заметите, как глубоко вы зашли.

Из-за отсутствия знаний о предстоящих невзгодах у меня не образовалось в голове убеждения, что сейчас мне будет трудно. Я полез в болото, о глубине которого ничего не знал, а когда тина начала заливаться в рот – отступать назад было уже поздно, потому что тогда я приобрел еще одно важное убеждение: **пока у человека есть сложности – он счастлив**.

Это звучит противоречиво только в рамках современной культуры. В нынешнем мире за истину выдаются весьма разрушительные для нашей мотивации принципы, которые заключаются в том, что каждый из нас – уже прекрасен, а наше мнение – уже важно. Просто так, по умолчанию. Жизнь вокруг нас становится все проще и проще: охоту на диких зверей заменил поход в магазин; кровавую борьбу за право спариваться заменил брак или tinder; реальная физическая опасность в повседневной жизни нам грозит только тогда, когда мы случайно перешли дорогу на красный свет.

Человек не предназначен для такой жизни. Эволюция создавала существо, чей мозг не дает усидеть ему на месте до тех пор, пока он не удовлетворит свои базовые потребности. Но сейчас все наши потребности удовлетворены, а «трудности», на которые можно было бы потратить силы, трактуются современной культурой как некий рудимент существования, которого нужно избегать.

Мы были созданы для трудностей и развития. Борьба и преодоление препятствий – это естественное состояние человека. Наш мозг вырабатывает психическую энергию в невероятных количествах, и

если раньше она направлялась на выживание, то сейчас мы распыляем ее на незначительную и откровенную ерунду, тратя силы на ругань в Интернете, на потребление мусорной информации, на абсурдный культ продуктивности или же на очень поверхностное, неглубокое творчество, потому что для достижения глубины и приобретения навыков нужно превозмогать сложности.

Без преодоления у нас нет никакой причины высоко себя оценивать, а ведь именно самообесценивание – это одна из болезней современного мира. Хотите поднять самооценку и не дать ей упасть, когда вы обернетесь на свое прошлое и увидите, что не достигли того, о чем мечтали, будучи ребенком? Тогда учитесь и работайте.

Учиться создавать игры в одиночку – трудно. Сейчас, как вы уже знаете, инструменты для разработки игр стали невероятно простыми, и лучший способ вернуться в «первобытное» состояние и столкнуться с трудностями – это разрабатывать игры без поддержки команды, а учиться – без поддержки преподавателя.

Учиться будет трудно. Но справляться с трудностями – это естественное состояние человека. Ваш мозг постоянно выделяет психическую энергию именно для того, чтобы вы справлялись с трудностями. Без работы над собой вас ждет самообесценивание, влекущее за собой целую вереницу проблем и комплексов.

12. Зачем вообще делать игры?

Одна из причин, по которой мы можем забросить обучение и разработку игр в самом начале своего пути, заключается в неправильной трактовке целей, побудивших нас выбрать именно такую сферу деятельности.

Давайте лишний раз разберемся, правда ли вам нужно делать игры или же вы на самом деле хотите чего-то другого? Осознать свои истинные желания – дело отнюдь не из легких, и для понимания того, как зародилось стремление «делать игры», нужно еще немного углубиться в принципы работы нашего мозга.

Главным стимулом при генерации целей является удовлетворение наших базовых потребностей, которых на самом деле всего три: размножение, безопасность и статусность (иерархическая потребность). Не стоит думать, что наша психика отличается от психики пещерных людей. Мы все такие же дикари и волнуют нас на самом деле такие же вещи.

Как видите, потребности «делать игры» природой в человеке не заложено. Она появляется в голове, когда мозг инвентаризирует память с целью предотвратить риски. Мозг, анализируя все многообразие вероятностей вашего будущего, может обнаружить, что в вашей жизни есть риск, например, стать нищим. Такое неблагоприятное будущее напрямую повлияет на вашу безопасность – потому что не будет денег, чтобы лечить болячки, и на положение в обществе – потому что в нашем мире деньги и статус идут рука об руку; да, наверное, и удовлетворить желание размножаться тоже станет труднее, потому что вы будете заняты выживанием.

Мозг начинает искать способы предотвратить эти риски. Он определяет то, какой результат ваших предполагаемых действий спасет вас от предполагаемой нищеты, и приходит к выводу, что надо бы заработать денег. Именно заработок в приводимом мной примере становится потребностью. Дальше, инвентаризируя вашу память, мозг пытается найти информацию, способную посодействовать в формировании плана действий для достижения поставленной цели.

Например, когда-то в прошлом мы услышали пару фактов об успехе разработчиков компьютерных игр. Образ выдающегося творца виртуальных миров запомнился. Мозг отложил его на полочку и достал как раз в тот момент, когда обнаружил какие-то угрозы для наших потребностей. В поисках способа избавиться от обнаруженной им опасности он прикинул: «А может ли разработка игр минимизировать риск остаться неудовлетворенным в одной из ключевых потребностей?»

Вполне вероятно, что в вашем случае мотив делать игры появился по другой причине и из других воспоминаний, но сам процесс идентичен: мозг выдает вам мотивацию на какое-либо действие только в том случае, если он предполагает, что это действие закроет какую-либо потребность или хотя бы убережет от грядущих рисков.

Мозг рассуждает примерно так: «Сейчас что-то, кажется, грозит нашей безопасности – значит, нужно найти способ, как эту безопасность нам обеспечить. Так, посмотрим тут, поищем тут… ага, вот этот способ должен сработать. Дам-ка я своему носителю немного мотивации на реализацию именно этого способа, авось все получится».

В каком случае что-то может пойти не так? Например, если потребность закроется раньше времени, то выработка энергии закончится. Этот процесс работает следующим образом: вот вы увидели рычащего волка. Мозг тут же вытащил из памяти информацию об этих опасных существах, нарисовал перед вашим сознанием несколько картинок с кровавыми укусами; поставил вам цель – «надо скрыться от этого зверя», и выдал достаточно энергии, чтобы вы сорвались с места и бежали вдвое быстрее Томаса Берка.

Что произойдет, когда вы скроетесь от волка? Необходимость бежать и прятаться исчезнет. План действий станет бесполезным. Выдача психической энергии именно на быстрое передвижение – закончится.

Примерно по этой же причине люди считают эффективными различного рода курсы для обучения. Если вы занимаетесь самообразованием, то никакого конкретного «волка» у вас в комнате нет. А если вы заплатили за курс, то вашим «волком» станут преподаватель, сроки и риск потратить деньги впустую.

Ваш мозг нарисует тревожную ситуацию «исключения» из образовательного учреждения в случае, если вы будете учиться

недостаточно усердно.

Курсы искусственно создают вам потребность «не попасть в неприятную ситуацию», и тревога, которую вызывают нелицеприятные картины вашего будущего, где вы унизились перед группой, потеряли деньги или столкнулись с гневом учителя, вынудит ваш мозг генерировать достаточно мотивации.

При обладании знанием о причине, почему нужны курсы, наша современная культура начинает казаться чем-то, что полностью уничтожает наш потенциал: нас учат стремиться к стабильной работе, к жилью в безопасном районе, к надежным и верным отношениям.

Мы живем в таком изобилии, что не нужно быть «высшим классом», чтобы потребность в безопасности и размножении была удовлетворена. Вероятность, что вас съест дикий зверь, почти равна нулю, а вероятность умереть голодной смертью исключается повсеместным наличием магазинов.

Все наши базовые потребности способна закрыть весьма средняя зарплата. Мы перенасыщены изобилием, а без потребностей мозг не генерирует мотивации. Когда мы находимся в полном покое, комфорте и достатке, нам тяжело начать что-то делать, потому что у нас, как и у любого биологического организма, абсолютно пропадают всякие причины как-то менять свою жизнь.

Необходимость размножаться на некоторое время удовлетворяется на всем известных интернет-ресурсах, а потребность приобрести достойное место в иерархии себе подобных многими из нас достигается путем принижения талантов и качеств окружающих.

Общаясь с огромным количеством разработчиков, я заметил интересную тенденцию. Нас всех условно можно разделить на тех, кто преуспел на поприще создания игр, и на тех, кто годами топчется на месте. Все, разумеется, познается в сравнении, но в данном контексте я отнесу лично себя к «преуспевшим», потому что за 6 лет в индустрии я выпустил 3 разработанных в одиночку игры; 5 игр выпустил на Nintendo Switch как издатель, да еще, как видите, книгу написал. Наблюдаемая мной тенденция заключается в том, что и я сам, и те ребята, которых я считаю куда продуктивнее и успешнее себя, редко начинали свои карьеры, находясь в «зоне комфорта». Жизненные ситуации, побудившие нас начать саморазвиваться, едва ли можно назвать счастливыми.

Я слышал историю разработчика, который с грустью вспоминал, как ему не хватало денег на покупку сладостей жене; другой делился опытом проживания с наркозависимыми, потому что другого жилья он позволить себе не мог, а третий рассказывал про по-настоящему опасные ситуации, связанные с его не очень-то и легальной, но единственной доступной в его родном городе работой.

Мне самому пришлось пережить весьма длительный период, в который я был буквально нищим: я работал в центре временного содержания несовершеннолетних и за сущие гроши возился с трудными подростками и их куда более трудными родителями. Я сталкивался с детской наркоманией; с долгими и болезненными процессами лишения родительских прав; я разбирал и документировал события, о которых вы, может быть, слышали в новостях. В ночные смены полиция привозила в наше отделение убежавших из дома детей, чтобы на следующий вечер к нам явились их проспавшиеся отчимы и вперемешку с руганью в адрес своих чад демонстрировали тотальную невменяемость: один из них через каждое бранное слово отвлекался, успокаивался и предлагал мне купить у него металлолом.

Моя реальность состояла тогда из грустных историй о несчастных детях. Мне не верилось, что мои потребности могут быть удовлетворены. Спать удавалось очень мало, потому что частенько приходилось оставаться на ночные смены, а помимо работы еще была учеба в аспирантуре; я не мог позволить себе сходить к стоматологу или вообще вылечить хоть что-нибудь, если вдруг заболею; я не мог обеспечить будущего своей жене; я часто пропускал обеды на работе, потому что не хотел тратить деньги.

Когда у моего отца был день рождения, мне сложно было придумать достойный подарок, на который бы хватало средств. Он любил смотреть фильмы с ноутбука, а не со своего небольшого телевизора, так что я ему подарил… HDMI-провод. На день рождения родному отцу мне удалось подарить только чертов провод. Именно в тот момент мой мозг осознал, что я нахожусь в весьма плачевном положении. Он начал искать способы привести мою жизнь в порядок. Мозг выдал мне столько мотивации, что через какое-то время я смог на очередной день рождения отца принести ему в комнату телевизор с отличным экраном в 70 дюймов. HDMI-провод шел в комплекте.

Пока наша культура побуждает нас стремиться к комфорту и к состоянию, когда мы можем сказать: «У меня все нормально», мозг наш требует стресса и тревоги. Люди **платят** за то, чтобы над ними стоял серьезный преподаватель. Люди отдают деньги, чтобы их образованию сопутствовали экзамены и жесткие сроки сдачи работ. Вместо обучения дома на уютном диванчике многие предпочтут быть привязанными к графику занятий, потому что мы понимаем: вероятность получить навыки увеличится.

Нужда завести себе надзирателя-преподавателя доказывает необходимость наличия тревоги для появления мотивации. Если ваши задания будет проверять живой человек – вам будет страшнее сделать что-то неправильно, и именно этот страх обеспечит вас силами на то, чтобы вообще что-то сделать.

Даже если вы способны учиться самостоятельно и работать в одиночку, то представьте, сколько всего бы вы еще смогли добиться, если б от успешности вашей деятельности зависело бы нечто бóльшее, чем просто получение новых знаний. Я бы написал эту книгу за неделю, если бы все семь дней надо мной стоял серьезный мужик в маске и держал бы у моего виска заряженный пистолет, курок которого он готов спустить, если я еще раз, еще хоть раз отвлекусь на сообщения в VK.

Четко определите, какую потребность вы пытаетесь закрыть разработкой игр. Эта потребность существует – вы же заинтересовались этой книгой. Зачем вам все это? Вам просто нужен статус «разработчика»? Если да, то поверьте, вы его получите на первой выставке игр, куда принесете демонстрировать свой незаконченный проект. Доводить игру до полноценного выпуска в таком случае у вас не будет нужды. Вам нужны деньги? При таком раскладе вы вполне можете отвернуться от собственного проекта, когда вам подвернется какая-нибудь прибыльная работа или если вы найдете инвестора. Может, это звучит иронично, но отсутствие возможности устроиться на доходную работу станет в вашем нелегком деле только преимуществом. Скажу честно, разработчиков-одиночек очень неохотно берут в крупные команды, потому что все понимают, что мы привыкли все делать сами и сработаться с нами сложно.

Я рекомендую на любой из сходок спросить понравившихся вам разработчиков: «А зачем/почему вы вообще решили делать игры?»

Некоторые ответы блеснут оригинальностью, некоторые рассмешат, некоторые вдохновят, а некоторые обнажат личностные проблемы человека, с которым вы ведете беседу.

Например, на одной конференции я общался с парнем, который начинал разработку игры в команде из пяти человек, затем сократил ее до двух человек, а на самом мероприятии присутствовал вообще в гордом одиночестве. Его главной мотивацией к разработке игры было «послать к черту как можно больше народа и научиться делать все в одиночку». Из более близкого общения с ним я сделал вывод, что он долгое время был лишен возможности доказать окружающим свою самостоятельность и именно потребность получить статус независимого индивида давала ему стимул обучаться во всех направлениях сразу и сокращать свою команду.

Самый простой способ разобраться, зачем же лично вам делать игры, заключается в том, чтобы задать себе вопрос «зачем/почему?» пять раз. Такой нехитрый трюк очень помогает наладить и другие аспекты своей жизни, но мы тут говорим только про разработку, верно? Главное – не рационализировать эти ответы, не отвечать чужими словами, не жить чужими целями и не спешить; уже на третьем вопросе станет очень сложно, поверьте. Иной раз хватит и двух/трех ответов, чтобы понять, является ли разработка игр действительно тем, что вы хотите.

Для примера я задам этот вопрос себе. Почему я делаю игры? Потому что мне нравится рисовать, придумывать персонажей и создавать миры. Зачем мне рисовать, придумывать персонажей и создавать миры? Затем, что я хочу самовыразиться. Зачем мне самовыражаться? Так я стану счастливее. Зачем мне становиться счастливее, чем я был? Затем, что я боюсь того беспросветного существования, которое я вел до того, как нашел свое призвание в разработке игр. Почему я боюсь вернуться к тому существованию? Этот страх уже вне моего контроля. Психическая энергия на потребность в безопасности выделяется без моего участия, а то, какой была моя жизнь до выпуска первой игры, буквально грозило моему здоровью.

Как видите, ответы на пять вопросов в итоге свелись к тому, что я хочу удовлетворить одну из трех базовых потребностей. Выходит, что цель, с которой я делаю игры, никуда не денется – потребность в

безопасности останется со мной навсегда, а мозг до конца дней будет искать способы закрыть риски для моего здоровья.

Найти более простой путь для обретения желаемой безопасности можно забившись, например, в бункер, где можно будет питаться консервами и жить долго и скучно. Но сделать это нам мешает уже культура нашего общества. Именно она преображает наши примитивные потребности в нечто сложное и запутанное вроде «создания игр».

Каждый из нас определенным образом трактует и саму культуру, и то, что определяется внутри нее под словом «безопасность». Лично я под этим понятием подразумеваю возможность хорошо питаться, получать достойное медицинское обслуживание, а также развиваться в качестве профессионала, ибо новые навыки позволят мне минимизировать риск остаться ни с чем в тот момент, когда прежние умения перестанут быть востребованными.

Культура способствовала тому, что мое примитивное желание приняло столь сложную форму.

Вот почему убеждения, воспоминания и понимание своих потребностей помогают сделать игру **сильнее**, чем навыки владения конкретными программами: мотивация и силы делать игры берутся именно из умения понимать себя и регулировать процесс своего мышления своими убеждениями.

У меня в памяти очень хорошо сохранился документальный фильм **Indie Game: The Movie**, рассказывающий о судьбе нескольких выдающихся игровых разработчиков, и когда у меня возникла нужда изменить образ своего существования, мой мозг вспомнил это замечательное кино и выдал мне мотивацию повторить путь главных героев фильма, а также узнать о профессии разработчика еще больше.

Чем дольше я разбирался в том, кто же разрабатывает видеоигры, тем чаще сталкивался с тем, что иной раз за выдающимися проектами стояли разработчики-одиночки, чью роль мне так легко было примерить на себя.

> **Задайте себе вопрос: «Зачем/почему я хочу делать игры?» и ответьте на него пять раз. Убедитесь в том, что выбранная вами деятельность действительно способна закрыть имеющиеся у вас потребности.**

13. Кто делает игры в одиночку?

Знание об играх, разработанных одним человеком и разошедшихся безумными тиражами, очень благоприятно скажется на мотивации для достижения нашей цели. Если мы хорошенько запомним этих талантливых людей, поиграем в созданные ими и хорошие, и плохие игры и получим от этих разнообразных проектов эмоции, то это поможет в момент упадка сил снова прийти в себя – мы вспомним, например, что живет на свете некий Дайсукэ Амаейя, которому удалось справиться с еще более суровыми испытаниями, нежели мы поставили сейчас перед собой.

Дайсукэ Амаейя – один из первопроходцев движения независимых разработчиков. Еще в начале нулевых он в одиночку создал игру под названием **Cave Story**. В те времена еще не было крупных площадок для распространения игр, выпущенных без издателя. Издатели же, в свою очередь, занимались преимущественно крупными высокобюджетными проектами. Дайсукэ выложил Cave Story на своем сайте и позволил скачивать ее абсолютно бесплатно.

Cave Story представляет собой нелинейный платформер в духе Metroid и рассказывает крайне трогательную историю от лица небольшого робота. В игре изумительные и проникновенные диалоги, запоминающаяся музыка, несколько концовок и очень много талантливых дизайнерских решений, разбирать которые можно на страницах отдельной книги.

Игра разлетелась по всему Интернету, ее перевели на множество языков, а позже, уже в сотрудничестве с издателем, Дайсукэ выпустил Cave Story+, которую можно найти как в Steam, так и на игровых приставках.

Но именно оригинальная, бесплатная Cave Story перевернула сознание многих разработчиков. Еще бы: этот человек продемонстрировал то, как далеко лежат лимиты разработчика-одиночки. Вдохновившись примером Дайсукэ, Тоби Фокс в одиночку сделал игру **Undertale**, размер и активность фанатского сообщества которой поражает воображение.

Чтобы оценить популярность Undertale, я рекомендую использовать такой сервис, как steamspy.com. С помощью своих алгоритмов он может рассчитать примерное количество владельцев игры в Steam. Так, вы увидите, что у Undertale около 3 миллионов владельцев. Далеко не все крупнобюджетные игры расходятся таким тиражом, а у нас идет речь про небольшую игру, сделанную «на коленке» одним человеком! Игра подкупила аудиторию своей необычностью: например, в ней можно было закрутить роман со скелетом, а многие персонажи и вещи оказались далеко не тем, чем представлялись игроку в самом начале.

Если вас пугает минималистичный внешний вид Cave Story и Undertale, то я снова упомяну разработанную одним человеком игру под названием **Bright Memory**, которая на момент выхода блистала современной графикой и использовала систему трассировки лучей, создающую эффект реалистичных отражений на поверхностях. Кто-то (например, я) может счесть такое нагромождение эффектов вопиющей безвкусицей, но невозможно отрицать статистику, демонстрирующую более одного миллиона продаж.

Bright Memory выглядит и играется как высокобюджетный боевик от третьего лица с кучей динамичных сцен, взрывов и эффектов. Низкий бюджет игры и факт, что игру сделал один парень из Китая, выдает только небольшая продолжительность этого проекта.

Миллион проданных копий – это, конечно, великолепный результат. Возможно ли превзойти его, например, в десять раз? Определенно! **Stardew Valley**, которую Эрик Барон разрабатывал на протяжении пяти лет, была продана тиражом более 10 миллионов копий только в Steam. А ведь купить игру можно еще и на приставках и развивать свою виртуальную ферму уже там – именно вокруг созидания строится игровой процесс Stardew Valley.

Нельзя не упомянуть трогательную историю **To The Moon**, созданную на движке RPG Maker разработчиком-одиночкой по имени Кан Гао. Его грустное и светлое творение доводило до слез самых суровых и черствых игроков, а позже To The Moon стала целой серией игр, общий тираж которых воистину огромен.

Отдельного внимания заслуживает игра **Icono-clasts**, разработчик которой (Йоаким Сандберг) превратил свой проект в долгострой – он разрабатывал свою игру больше десяти лет, собрав все возможные

ошибки разработчиков-одиночек, но преодолел все моральные и материальные трудности. Его игра хоть и не стала многомиллионным хитом, свое признание в определенных кругах она получила и является отличным представителем независимых игр.

Существует еще множество проектов, которые могут вас заинтересовать и вдохновить: The Stanley Parable, Wings of Vi, The Beginner's Guide, Return of the Obra Dinn, Dwarf Fortress, Papers Please, Five Nights at Freddy's, Pinstripe, Dust: An Elysian Tail, Thomas Was Alone, LISA, Axiom Verge, Figh'n'Rage, Mystik Belle…

Я чувствую, что скептики уже запаслись аргументами против моей позиции о невероятных возможностях человека. Главный из этих аргументов заключается в том, что все перечисленные мной игры можно отнести к «ошибке выжившего»: не стоит судить об успешности разработчиков-одиночек по проектам, которые выстрелили и стали сверхуспешными. Существуют тысячи игр от разработчиков-одиночек, которые провалились в пух и прах и о которых мы ничего не знаем.

Я не буду с этим спорить: одиночек, игры которых провалились, в сотни раз больше, чем тех, кто выпустил успешный проект и исчисляет количество проданных копий в миллионах.

Но не говорите, что с командами разработчиков дела обстоят иначе. Тысячи проектов, созданных усилиями как больших студий, так и маленьких команд, провалились. Только вот критерии провала для игры, разработанной в одиночку, отличаются от критериев провала игры, разработанной дружной компании ребят: они-то делят прибыль между собой.

Если игра, разработанная в команде из пяти человек, продалась тиражом в сто тысяч копий (что очень и очень хорошо для игры от независимой студии), то каждый из разработчиков получил доход с двадцати тысяч копий. Если же я один продам сто тысяч копий, то я получу доход от ста тысяч копий. Наличие еще четырех человек в моей «команде» не сделает игру ровно в четыре раза успешнее или в четыре раза лучше. Скорее, наоборот…

Игре от разработчика-одиночки гораздо легче окупиться.

И, перед тем как мы перейдем уже к составлению программы действий, я хочу еще раз вспомнить Сакагучи и Миямото. Многие современные независимые игры, ставшие хитами, действительно

близки по техническому исполнению к играм прошлых эпох. To The Moon вполне могла бы существовать на SNES, а Iconoclasts выглядит так, словно бы это игра от SNK с аркадного автомата из середины девяностых годов. Вот только Кан Гао не писал своего рендера, а Йоаким Сандберг не заморачивался о количестве спрайтов и размере уровней. Современные технологии выполнили за этих разработчиков ту работу, с которой раньше справлялась команда людей.

Обязательно поиграйте хотя бы в одну игру, разработанную соло. Так вы сформируете представление о возможностях разработчика-одиночки и узнаете о человеческих возможностях еще чуточку больше.

14. Какой порядок действий-то?

Итак, мы определились с тем, какие риски и невзгоды придется преодолеть и какие потребности требуется закрыть.

Стало очевидно, что именно разработка игр удовлетворит наши нужды. Нам удалось запастись не только убеждением, что мы созданы для сложностей, но и примерами из жизни людей, которые преодолели непреодолимое и показали лимиты человеческих возможностей.

Но в этой мозаике есть еще один элемент, который может помешать довести проект до конца: вы можете знать, чего хотите; вы можете быть убеждены, что для достижения желаемого нужно именно сделать игру, притом исключительно в одиночку. Но только вот тропа, которая лежит между вашей мотивацией и вашей целью, едва видна в густом лесу из вопросов: «А за что хвататься-то? Как делать игры? Как начать?»

Если вы все еще не осознаете конкретную программу действий для самообучения, то вот она, простая и прямолинейная.

• Выбираем движок.

• Находим абсолютно любой курс или набор уроков в сети и начинаем кропотливо повторять все, что нам вещают с монитора. Я не рекомендую привередничать в выборе автора уроков в самом начале, потому что вы все равно не располагаете достаточными знаниями о движке и не сможете здраво оценить компетентность человека, которого слушаете.

Просто будьте внимательны к собственным ощущениям и к результатам своей деятельности: если вам ничего не понятно, если все работает не так, как рассказывает автор уроков, если вам не подходит темп повествования или не устраивает длина роликов – ищите другую серию занятий. Их много. У вас есть из чего выбирать.

Я уверен, что если вы уже зашли так далеко и принялись за учебу, то у вас есть хоть какая-то идея игры, которая определит направленность уроков: будет это платформер или ужастик от первого лица? Искать ли вам уроки про работу с 2d на Unity или про систему передвижения в Unreal Engine? Если же идей нет, то наберитесь пока терпения, скоро мы поговорим о способах придумывать игры.

• Освоив интерфейсы, ознакомившись с возможностями движков, попробуйте сделать в стол хотя бы что-нибудь авторское и самостоятельное. Можете попытаться повторить механику из игры, которая вам нравилась в детстве.

Под «механикой» подразумевается набор условий и правил, используемых в игре. Вид от первого лица + возможность стрелять во врагов – это механика шутера. Необходимость добраться от точки А до точки Б с «видом сбоку» и с необходимостью постоянно прыгать – это механика платформера. Самих механик может быть множество. Ничто не мешает в механику платформера добавить механику прокачки из ролевых игр. Продумывать взаимодействие механик друг с другом – очень увлекательный и сложный процесс, но в начале пути я рекомендую остановиться на простых и популярных механиках, не создавая никаких нагроможденных симбиозов из гонок и шутера.

Тут вам на помощь уже придут не серии уроков и курсы, а конкретные запросы для решения конкретных проблем в духе «как сделать исчезающую платформу в GameMaker». Поисковые системы выведут вас или на короткие видео по теме, или на обсуждения на форумах. А если нет, то вы всегда самостоятельно можете задать любой вопрос на одном из посвященных разработке форумов.

• Вам придется рисовать или моделировать в том случае, если вы хотите создать авторскую игру, а не склеить ее из ассетов. Подход в обучении рисованию никоим образом не отличается от подхода к изучению движка: садитесь, ищете уроки, повторяете, повторяете, повторяете. Поверьте, вам не нужно достигать академического совершенства, вы вполне можете придумать проект, основанный на ваших навыках. Например, сейчас становятся популярными игры, выполненные в стилистике игр PS1 и демонстрирующие весьма простые модели персонажей.

В плоском мире 2D-графики возможностей упрощать себе жизнь с помощью выбора удобного стиля тоже неимоверно много. Я рекомендую вам посмотреть на игру Limbo, графика в которой не блещет деталями, но запоминается своей мрачностью и минимализмом, а также стоит приглядеться к проекту **OneShot**, который при достаточно лаконичном исполнении радует пользователя стильной картинкой за счет простеньких световых эффектов и приятной цветовой палитры *(рис. 10)*.

Хорошая палитра должна стать главным центром внимания начинающего 2D-художника, потому что иной раз только благодаря ей игра может обрести привлекательный вид. Представьте, если бы в **Thomas Was Alone** или **Limbo** был кислотно-зеленый фон и розовые персонажи. Строгий и тонкий вкус вмиг бы растворился в месиве из дурно сочетающихся цветов.

Рис. 10. ***Limbo, 2011 год. Игра, изящная в своей лаконичности***

В создании палитр вам помогут бесплатные веб-приложения, (например, **Adobe Color**), пресловутые уроки на YouTube, а также работы других художников. Я рекомендую вырабатывать «насмотренность» путем сохранения тех изображений, стиль и палитра которых вам приглянулись. Не стоит заниматься бездумным накоплением «прикольных картинок» или видеороликов. Вам нужно анализировать то, что вы видите. Пытайтесь мысленно ответить себе на вопросы: как художник реализовал переходы между цветами на этой картинке? Что здесь за палитра? Как устроена композиция? Чем больше материала вы проанализируете, тем осмысленнее вы будете созидать, потому что развитый вкус – это основной залог создания радующего глаз изображения. Используйте трюки, подсмотренные у

других художников, и осознайте свои собственные предпочтения. Нравится ли вам обводка у персонажей? Нравится ли вам аниме стилистика, или западный стиль рисования находит в душе бóльший отклик? Используйте сочетания цветов, которые приглянулись; считайте, сколько видов деревьев нарисовал художник для игры, которая вас зацепила своим внешним видом; обращайте внимание на то, как дизайнер расположил спрайты и в чем была особенность их рисования… Это бесконечный и очень увлекательный процесс!

Еще один неплохой трюк заключается в том, чтобы постоянно перерисовывать одно и то же. Если первой вашей работой стало, например, дерево (3D или 2D – не имеет значения), вряд ли вы сразу придете к выводу, что оно шикарно настолько, чтобы вводить его в игру. Скорее всего вы приметесь «полировать» детали, крутить ползунки и пытаться привести свою неуклюжую и некрасивую работу в божеский вид «малой кровью» – не перерисовывать же по новой, верно?

А вот и неверно. Перерисовать с нуля то, что вам не нравится, всегда эффективнее и быстрее, чем пытаться исправить рисунок, загубленный еще на стадии наброска. Нарисуйте или смоделируйте один и тот же объект несколько раз с нуля – поставьте эти рисунки рядом – и вы очень сильно удивитесь.

Я постоянно наблюдаю за тем, о какие вещи спотыкаются разработчики в своих начинаниях. Одна из наиболее частых ошибок заключается в том, что работа новичка больше напоминает некое подобие мышиной возни: он слишком сконцентрирован на деталях своего проекта и не может поднять голову, чтобы понять направление своего движения.

Если привести совсем конкретный пример, то я во время собственного обучения рисованию пытался изобразить Женщину-кошку из Вселенной комиксов Marvel. Как бы я ни старался – рисунок получался вялым и невыразительным. Чуть позже я осознал, что моей ключевой ошибкой стала концентрация на мелких деталях и отличительных признаках выбранного мною персонажа: вместо того чтобы набраться терпения и разобраться с тонкостями черт лица Женщины-кошки, определиться с расположением губ, размером глаз и длиной носа, я уделял колоссальное внимание ее стильной маске. Маска, безусловно, является очень ярким элементом, но концентрация

на такой мелочи на ранних стадиях превращала мою работу в неуклюжую возню. Маска хоть и была одним из элементов, характеризующих персонаж, но она всегда должна была оставаться лишь дополнением к общей картинке.

Первое время персонаж должен походить на набор шариков, конусов и цилиндров, и уже после того, как в этих грубых кусочках проявится читабельный образ, наступит время, чтобы низко склонить голову над листом бумаги и наконец-то начать уделять время деталям.

Что касается самой разработки игр, концентрирование на мелочах выглядит немного иначе. Я знаком с разработчиком, который несколько месяцев топтался на месте и не мог заставить свою игру быть увлекательной. Его система боя казалась неудобной, некомфортной и, что самое страшное – непонятной. Вместо того чтобы взглянуть на картину в целом и понять фундаментальность своих ошибок, он пытался исправить общее улучшением частного. Иными словами, вместо того чтобы переделать неработающую систему боя целиком, он пытался добавить больше эффектов, больше элементов интерфейса, поменять местами пару иконок и надеялся, что это поможет.

Хорошая игра весело играется и на «кубиках» – без красивых интерфейсов или спецэффектов. Если что-то вас смущает в собственной идее еще на стадии совсем сырого прототипа, то поверьте, нагромождение на этот прототип мелких деталей и красочных спецэффектов лишь усугубит ситуацию. То, что вам кажется несовершенным в самом начале, при дальнейшей обработке будет становиться только хуже.

Страсть заморачиваться на «детальках» в самом начале пути мне ясна и понятна: мы стремимся получить как можно больше эмоций от своей деятельности, а эмоции нас захлестнут уже тогда, когда перед глазами мы увидим хотя бы малюсенький «готовый» кусочек. Ведь вряд ли может обрадовать то, что месяц разработки увенчался всего лишь прыгающими по экрану кубиками. Мы же еще не понимаем в начале, что получится в итоге, а понимание это нам нужно для получения удовольствия от своей работы.

Но, чем больше у нас накапливается опыта, чем больше мы прошли путей от сырого наброска к готовому рисунку или от кривого прототипа к завершенной игре, тем проще нам будет обрести

необходимый для разработчика навык – видеть готовое в незавершенном.

Когда опытный художник смотрит на грубый набросок, он способен увидеть всю картину целиком. Когда опытный разработчик видит прототип с кубиками – он может представить, как эта механика будет смотреться, когда появятся интерфейсы и графика. Если вам сейчас показать грубый скетч гениальной картины, вы не поймете, как таким результатом можно наслаждаться – тут же ничего не понятно! Но автор в грубых и неровных линиях видит красивую и почти завершенную работу.

Приобрести умение «мысленно дорисовывать» созданное можно не только путем проб и ошибок. Тем, кто рисует, я очень рекомендую просмотр видеороликов из разряда Speed Painting – они представляют собой ускоренную запись того, как художник создает то или иное произведение. Обратите внимание, в какой момент происходит переход от общего к частному, и пытайтесь выработать навык не строить свою работу вокруг одной-двух деталей.

Рис. 11. ***Очень ранний концепт игры Ori and the bling forest***

Для игрового дизайна я же рекомендую посмотреть видеоролики с бета- и альфа-версиями знакомых вам игр. Найти эти ролики можно по запросу, включающему в себя название игры и подпись alpha, demo или concept. Таким образом тем, кто играл в **Ori and the Blind Forest**, будет интересно посмотреть на YouTube видеоролик Ori and the blind forest Interaction Concepts *(рис. 11)*.

Несмотря на чудовищно примитивную графику, даже этот ролик выглядит как демонстрация весьма интересной и динамичной игры: кубик, которым мы управляем, ощущается эластичным и подвижным, а происходящее на экране вызывает желание взять в руки контроллер и попробовать попрыгать на этом тусклом фоне.

Но если уже на такой стадии играть в ваш проект неинтересно, будет ошибкой думать, что нагромождение деталей спасет игроков от уныния, в которое вы собрались их погружать. В хорошую игру будет интересно играть и без графики, частиц и наворотов.

Долгое время огромная часть вашей игры будет существовать только в воображении. Умение видеть в своих набросках целые миры, полные

деталей и нюансов, – очень важный навык, который способствует получению светлых эмоций как во время разработки, так и на этапе обучения. Нужно радоваться тому, что сегодня вы смогли заставить квадратик на экране прыгать, потому что в вашем воображении этот квадратик уже давно принял облик интересного героя, а серый фон – это чертовски красивый лес, призывающий игроков отправиться в неповторимое приключение.

Сильные эмоции во время процесса обучения и созидания – важнее, чем кажутся.

Для того чтобы определить место эмоций в столь, казалось бы, изнурительной деятельности, как самообразование и разработка игр, нам снова нужно поговорить о чем-то очень важном – о нашей памяти.

Всегда больше думайте об общем, а не о частном. Не увлекайтесь деталями до тех пор, пока общая картинка не станет ясной.

15. Я все забыл

Процесс обучения работе с Blender, Photoshop, Unreal Engine, Construct или с любым другим приложением можно условно разделить на две стадии: сначала мы осваиваем возможности программы и обретаем понимание того, **как** в ней что-либо созидать, а в ходе второй стадии мы нарабатываем навык и совершенствуем собственные умения.

Первая стадия может вызывать трудности и недоумение в первую очередь при попытке освоить игровые движки: для успешной работы вам нужно научиться разговаривать с ними на их языке.

Процесс создания какой-либо игровой механики выглядит примерно так: сначала механика появляется у вас в голове (например, вы хотите, чтобы платформа начинала падать вниз спустя секунду после того, как игрок на нее приземлился), а потом вы пытаетесь **объяснить** вашу идею движку. Для «объяснения» вы уже используете язык самого движка, логика которого варьируется от программы к программе. Разными способами в разных приложениях вы будете растолковывать, какая конкретно платформа должна упасть после того, как ее коснется персонаж.

То же самое касается и рисования: вы должны научиться объяснять программе, какого цвета и какая кисть вам сейчас нужна. На первой стадии обучения вы осваиваете **логику** программы. Вы постепенно начинаете понимать, какая кнопка в интерфейсе программы вызовет ту или иную реакцию. Вы придете к пониманию, что произойдет, если вы напишете выражение вроде «lerp (Variable1, 1, 0.1)» (в Construct 2 это будет означать, что значение переменной, которой вы дали название Variable1, станет «1» за 0,1 секунды, пройдя все тысячные доли от значения, которому Variable1 была равна до запуска этого выражения).

Кроме логики для освоения новых программ нужна, разумеется, и **память**. При просмотре урока, например, по GameMaker вам может показаться понятным каждое действие преподавателя, но, когда вы сядете воплощать в жизнь то, что вы только что видели, к вам быстро придет осознание, что вы не запомнили ни порядка действий, ни расположения кнопок.

Знания – это не какой-то ресурс, который можно накопить, а потом использовать в нужный момент. Мозг не хранит все, что мы видели. Бытует очень вредный миф о том, что в нашей голове отпечатывается любое, даже самое незначительное событие. Может, так оно и есть, но вот «достать» это воспоминание – непосильный труд. «Под рукой» мозг держит только тот набор воспоминаний, который, как он считает, поможет ему сделать его главную работу – минимизировать риски и обеспечить ваше выживание.

Лучше всего запоминается то, что хотя бы слегка отличается от «нормы». Попробуйте вспомнить то, чем вы занимались в прошлом месяце. Какие-то конкретные дни. В первую очередь в памяти всплывут какие-то необычные события, вроде прогулки по новому месту или встречи с новыми людьми, а вот вспомнить «обычный вторник» будет уже сложнее. Мозг не хранит информации о ваших действиях и событиях вокруг вас, если эти действия и события мало отличаются от уже знакомого опыта. Трудно вспомнить каждую поездку на работу. А вот если в один из дней по дороге встретилась ваша одноклассницу, которую вы не видели десять лет, то такое событие запомнится.

Мы запоминаем те элементы, с которыми мозг связал или отрицательные, или положительные эмоции. Когда события протекают «ровно» и без эмоций – они очень быстро забудутся.

Если мы получили удовольствие – мозг это запомнит, чтобы потом повторить. Если мы получили заряд негативных эмоций – мозг это запомнит, чтобы, напротив, избежать повторения.

Во время разработки игр я сталкивался с тысячей проблем и исправил тысячи багов в своих играх. Помню ли я каждый баг? Конечно, нет. Я отлично запомнил первый баг, мешавший игрокам наслаждаться моей первой игрой, потому что такая деятельность, как исправление ошибок, была мне в новинку и выделялась среди тех дел, которыми я занимался обычно. Помимо этого, я отлично помню совсем сумасшедшие глюки, которые превращали мои проекты в цирк с конями и полностью ломали правила игры. Это отложилось у меня в памяти по той причине, что я испытал очень яркие эмоции, когда увидел от своего тестировщика видео, где он вытолкнул игрового персонажа с игрового поля и оправил его гулять за границами экрана.

Как вся эта информация о нашей памяти поможет в обучении? Вы должны понять, что нужно как можно скорее пытаться применить полученные знания. Взгляд на нечто, что вы создали сами, очень воодушевляет. Если мозг зафиксирует, что вы испытали радость при взгляде на свой рисунок или 3D-модель, он обязательно сохранит информацию о тех шагах, которые вы сделали для достижения яркого результата.

Другой пример использования этого знания я наблюдал, когда, еще будучи бакалавром, преподавал компьютерную грамоту пожилым людям. Я рассказывал им про то, как перетаскивать файлы из папки в папку и чем «вырезать» отличается от «копировать». Я видел скучающие лица и понимал, что моя группа едва ли запомнит текущую тему – никто здесь не испытывал ярких эмоций. Тогда я перескочил через урок и научил их менять заставку рабочего стола. Они тут же увидели явный результат приложенных усилий и впали в детский восторг: их действия привели к тому, что на рабочем столе меняются картинки! «Это весело, это приятно, это надо запомнить», – решил тогда их мозг и «захватил» с собой еще несколько событий, предшествующих смене обоев для рабочего стола, а именно пресловутую тему про «вырезать и копировать».

Еще множество лет назад психологи выявили, что обучение происходит быстрее, когда новая модель поведения, который вы только что обучились, моментально приводит к обратной связи: например, к резкой смене изображения на рабочем столе.

Первые часы обучения работе в Blender сводятся к манипуляции с кубом, что не вызывает никаких эмоций, потому что это крайне скучный процесс. А вот когда руки дойдут до кнопки, добавляющей к вашему кубу загадочные «модифаеры», то первое время вам будет очень весело перебирать все возможные комбинации, превращающие ваш подопытный кубик то в бублик, то в шарик, то в ромбик. Ваш мозг сохранит воспоминание о «модифаерах».

В чередовании «знание – результат – знание – результат» и заключается качественность курса. Если первые 20 уроков вам будут просто объяснять, как работают интерфейсы и не предоставят демонстрацию того, что же вы можете создавать в этой программе, то запомнить что-либо из услышанного у вас получится с большим трудом.

Стремитесь получать эмоции от образования. Демонстрируйте ваш результат близким, чтобы получить удовольствие от их гордости за вас и тем самым лучше запомнить способ, с помощью которого вы этого результата добились.

Разбавляйте скучные технические уроки обучением чему-то, что или зрелищно, или находит отклик именно в вашей душе. Я восхищаюсь женской красотой и начинал обучение рисованию именно с женской анатомии, чтобы как можно быстрее получить на холсте те контуры и образы, что вызывают у меня восторг.

Всегда повторяйте уроки за виртуальным преподавателем прямо в момент чтения урока или просмотра видео. Множество исследований доказывает, что применение знаний сразу после получения многократно увеличивает срок хранения этого знания в вашей голове.

Без осмысления и повторов любое знание улетучится, не оставив и следа.

И помните важную вещь: пока вы читаете текст или смотрите видео, мозг потребляет информацию, а не усваивает ее. Переработка потребленной информации начинается в моменты «простоя» вашего думающего мозга, т. е. тогда, когда вы не заняты вообще ничем и ни о чем не думаете. В наше время люди неосознанно позволяют мозгу простаивать только во время сна. Именно тогда воспоминания и знания раскладываются по полочкам и усваиваются.

Перерывы в обучении нужны не только для того, чтобы сохранить силы, но и чтобы лучше запомнить изучаемый материал. Перерыв не должен строится на логике «отвлекусь на что-нибудь другое, например, проверю социальные сети». Вы так не дадите вашему мозгу отдохнуть, потому что для него на самом деле нет никакой разницы, что за информацию вы пытаетесь обработать – документацию **Unreal Engine** или подборку смешных картинок. Мозг продолжает и работать, и запоминать.

Перерывы в процессе обучения должны сопровождаться информационным вакуумом. Прогулка без наушников; душ или ванна; просто восседание в середине комнаты в полной тишине – все это способствует процессу запоминания. Именно в эти моменты мозг обрабатывает полученную информацию.

Каким образом настроить свое мышление на то, чтобы в моменты «простоя» не пребывать в отвлеченных мыслях, станет вопросом

следующих глав, а пока мы вернемся к обучению. Если первая часть процесса обучения заключалась в освоении возможностей и логики программного обеспечения, с которым мы имеем дело, то следующая фаза заключается в освоении и совершенствовании своих собственных возможностей.

> **Применяйте полученные знания как можно быстрее. Демонстрируйте результаты своего обучения тем, кто может за вас порадоваться. Чтобы отдохнуть, не используйте социальные сети или просмотр фильмов – вашему мозгу нужно побыть в тишине и покое, чтобы обработать полученные знания.**

Вторая часть обучения становится доступной, когда мы уже более-менее научились понимать логику инструмента, с которым мы работаем. Мы уже можем «объяснить» ему какая кисть нам нужна в Photoshop, какую строчку кода надо выделить в Unity или куда переместить тот или иной объект в Blender. Теперь нам нужно нарабатывать **навык**. Один из лучших советов, что я слышал: «Лучше всех рисует тот, кто больше всех рисует». Конечно, стоит сюда добавить, что рисовать нужно осмысленно и самостоятельно выискивать ошибки, которые вы совершили, но тогда красота и лаконичность этой фразы немного развеется.

Против самообучения изобразительному искусству существует весьма сильный аргумент, и заключается он в том, что вам нужен мастер, способный указать на ваши ошибки и преподнести на блюдечке способы их устранения.

Я учился рисовать без надзирателя. Я знаком со многими 3D-моделлерами, у которых также никто никогда не «проверял» их работ. Тем не менее их труды выглядят замечательно. Для достижения такого результата необходима, в первую очередь, насмотренность, о которой я уже упоминал, но в контексте поиска ошибок в своих работах раскрою этот пункт чуть шире.

Мы с вами воспользуемся таким замечательным человеческим свойством, как неспособность видеть бревна в своем глазу и способность находить соринки в чужом. Смотрите не только на те рисунки, модели и игры, которые вам нравятся, но уделяйте не

меньшее внимание тому, что вызывает у вас отторжение. Вот вы наткнулись в сети на картинку и считаете ее полной ерундой. Задайте себе следующие вопросы: что не так с этой картинкой? Почему эта модель кажется вам такой некрасивой? Почему вам не нравится эта игра? Таким образом вы наберетесь знаний о тех вещах, которых стоит избегать в своем творчестве.

Вторым способом исправления ошибок в работах может послужить простая загрузка вашей первой картинки, модельки или игры в Интернете. Существуют группы начинающих художников и 3D-моделлеров, где вы можете запросить критическую оценку, а также почитать, за что ругают других. Квалификация тех, кто начнет вас критиковать, не всегда будет заоблачно высокой, и я не буду отрицать, что чаще всего академические знания у комментаторов будут отсутствовать. Но едва ли эти знания найдутся у тех, кому предстоит делать репосты ваших работ и покупать вашу игру. Потому их мнение все равно очень и очень важно.

Ваши работы должны нравится не академикам, а обычным пользователям.

Для того чтобы быстро тренировать навык, нам нужен мужик в маске, который будет держать пистолет у нашего виска и торопить нас производить столько картинок, чтобы планшет или мышь начинали дымиться, а кода писать столько, чтобы у клавиатуры вылетали клавиши. К нашему счастью, в мире разработки игр такой мужик существует. Он носит гордое имя Game Jam.

16. Игровое варенье

Сам термин Jam пришел к нам из музыкальной индустрии. Событие, когда участники группы собирались вместе, чтобы без предварительной подготовки создать новый материал или же просто потренироваться, называлось Jam Session. Game Jam подразумевает примерно то же самое: организаторы джема выбирают тему игры и устанавливают сроки (от двух дней до нескольких месяцев), а участники – как вы могли догадаться – делают на заданную тему игру.

Количество проводимых на просторах Интернета джемов – бесконечно. Иногда джемы организовывают с денежными призами, иногда призом является контракт с издателем, а иногда на джемы привлекают участников, не предлагая взамен ничего кроме веселья. Свою платную лицензию движка я выиграл на джеме от сообщества c2commnunity.

Рассмотрим, как выглядит джем на примере такого мероприятия, как **Ludum Dare**, – одного из крупнейших джемов, в котором участвуют разработчики со всех стран мира. Количество участников сводится к тысячам человек. Организация чаще всего выглядит следующим образом: сначала участники голосуют за тему игрового джема, затем объявляются даты, в которые он проводится, и прямо перед самым включением таймера объявляется тема, набравшая больше всего голосов. На разработку игры дается обычно два-три дня. Работать можно как в команде, так и в одиночку – для обоих способов участия предусмотрены разные сроки сдачи работ и разные номинации.

Во многих городах в это время проводятся мероприятия, которые собирают участников джема под одной крышей. На таком мероприятии можно познакомиться с другими ребятами, найти себе команду или просто посмотреть на то, как протекает творческий процесс. Иной раз организаторы предоставляют даже место для сна.

В последние минуты джема разработчики судорожно заливают сборки своих игр и оформляют странички на предназначенном для этого сайте. Чаще всего используется ресурс **itch.io**, на котором по запросу Ludum Dare вы можете ознакомиться с тем, что вообще представляют собой игры, сделанные так быстро. Чаще всего это

очень коротенькие проекты, в которых разработчики демонстрируют необычные идеи, механики и пытаются порадовать пользователя не степенью прорисовки и детализации объектов, а минималистичным стилем и необычным игровым процессом.

Прославленная игра **Hollow Knight** начиналась именно как проект на джеме. Моделька полого рыцаря, созданная за три дня, практически без изменений перекочевала в финальную версию игры, что разошлась многомилионным тиражом. Как проект на джеме начиналась и игра **Celeste**, доработанная версия которой собрала огромное количество наград и получила общественное признание. Джемы стали прекрасным способом, чтобы обкатать идеи этих игр.

После публикации проектов начинается голосование. У **Ludum Dare** в качестве жюри выступают сами участники – вы играете в проекты коллег и выставляете им оценки по разным шкалам вроде графики, настроения и увлекательности. Чем больше оценок вы проставили, тем больше оценок и отзывов вы получите в ответ. Пожалуй, нигде, кроме **Ludum Dare**, я не видел системы, позволяющей заработать такое огромное количество комментариев к своему творчеству. Именно по этой причине данный джем годится для того, чтобы опробовать свою идею и прикинуть, нравится ли она игрокам, насколько она им понятна и стоит ли продолжать делать игру в том же направлении.

Кому-то может показаться плюсом возможность быть замеченным. Например, мой проект на Ludum Dare, который я случайно засунул в номинацию игр, разработанных в команде, а не в одиночку, занял там 32-е место. Не самый плохой результат, но можно и лучше. Тем не менее мне написал издатель маленьких игр, который предложил выкупить у меня этот проект за небольшую, но очень приятную сумму.

Но важность джемов заключается не только в возможности найти коллег по цеху, обкатать механику своей игры и получить признание. Как обычно, самое ценное заключается во влиянии подобного мероприятия на наш мозг. Сейчас придется вспомнить то, что я говорил ранее про мотивацию и цели.

Мозг может определить наши неудовлетворенные потребности; он может отыскать в воспоминаниях информацию про разработчиков игр и интерпретировать ее как один из способов потребности наши удовлетворить. Но то, чего ему не хватает по сей день, так это **программы действий**. Логическая цепочка «я не доволен – сделаю

игру – стану доволен» не очень сложна, а вот если углубляться в конкретику, то точных шагов мы пока не знаем. Как эту игру делать-то? Даже если мы научились программировать, рисовать и насмотрелись на игровой дизайн в других играх – программы действий по применению этих знаний у нас пока еще нет.

Эту программу действий обеспечит нам игровой джем. Разработка игры за три дня отличается от разработки игры за три года только масштабами – делать нам придется то же самое и, возможно, в том же самом порядке. Когда мы разработаем небольшую игру за три дня, программа действий для разработки игры за пару лет станет нам гораздо понятнее. Качество самой игры даже не будет иметь значения. Разработка игры на джеме поможет определить наши слабые и сильные стороны и сформулировать, а каких навыков и знаний нам бы еще поднабраться?

На любом джеме я пытаюсь разбить работу на три равные части: сначала я занимаюсь механиками. Я придумываю, какие события развернутся в моей игре, какие препятствия встанут перед игроком и какими навыками их преодоления я его наделю. На разноцветных кубиках я реализовываю ключевых врагов, физику передвижения и возвожу какие-то элементы уровней.

Во второй части разработки я стараюсь разобраться со всей графикой в игре: нарисовать персонажа, анимировать его, добавить ему врагов и нарисовать блоки и спрайты, из которых я буду в дальнейшем собирать уровни.

Третья часть заключается в «натягивании» графики на код и в создании самих уровней. Из разработанных в первых двух частях материалов я уже собираю саму игру, словно конструируя полосу препятствий для игрока.

> **Участие в джемах выявит и ваши слабые стороны. А знание своих слабостей поможет вам спланировать разработку полноценной игры так, чтобы исключить из нее элементы, с которыми вы не справляетесь. Совсем не выходит писать диалоги? Так расскажите историю без слов.**

Ограниченное время, как ни странно, служит здесь моим союзником, потому что позволяет выработать опытным путем один из

важнейших навыков разработчика: отсекать свои идеи. Во время разработки игры должен наступить момент, когда пора перестать придумывать новые интересные штуки, прекратить вводить новых персонажей и больше не расширять игровую вселенную. Без навыка сказать себе «нет» в тот момент, когда в голове возникает идея «а давай добавим персонажу тройной прыжок!», разработка игры может превратиться в многолетний долгострой.

Рамки, как временные, так и творческие, как ни странно, являются отличным помощником креативности: творец без рамок погрязнет в разработке на долгие годы. Я могу потратить на анимацию главного персонажа хоть несколько месяцев, но я занимаюсь этим, чтобы игры делать, а не залипать в том, смысл чего весьма сомнителен: чем дольше мы делаем игру, тем больше вероятность отказаться от этого проекта вовсе, растеряв всякую мотивацию, цели и желания, или же просто застрять в бесконечных переделках.

Мы лучше учимся, если над нами есть надзиратель, и эффективнее работаем, если нам грозит штраф или увольнение. Мы – социальные животные, и отсутствие взаимодействия с другими людьми, как ни странно, не делает нашу жизнь проще. Без угроз, наград и надзирателей мы сами должны себя контролировать.

Но это так сложно. Давайте разберемся, почему.

17. Суровая дисциплина

Самодисциплина – единственный моральный принцип. Свобода – это не право действовать как душе угодно, а контроль над самим собой.

Pascal's Wager, надпись на каменном алтаре

Неверное, представление о дисциплине является одним из вредных убеждений, которыми наши «ядра» будут склонять нас к безделью. Образ дисциплинированного человека, нарисованный массовой культурой, чрезвычайно деструктивен: по моим наблюдениям, за дисциплинированных людей выдаются такие суровые ребята, которые встают в пять утра, принимают ледяной душ, завтракают овсяной кашей с камнями, вкалывают 12 часов без передышки и в восемь вечера ложатся спать на выглаженные простыни.

Превращаться в такого персонажа не хочется. Кажется, что подобные люди живут в вечных лишениях и крайне несчастны. Многие из нас сводят понятие дисциплины к игнорированию своих желаний: вам, например, хочется понежиться в теплом душе, но вы обязаны включать струю ледяной воды; вам хочется поиграть в видеоигры после работы, но вы вынуждены в очередной раз тратить время, например, на повторение того невыносимо сложного урока по Unreal Engine.

На самом деле дисциплина не связана с игнорированием своих желаний. Чтобы объяснить эту позицию, мне придется отвлечься и пояснить, что мы на самом деле постоянно существуем в трех временных периодах – прошлом, будущем и настоящем.

Мы мыслим прошлым, когда оцениваем какое-либо событие или поступок, исходя из полученного нами ранее опыта. Программа действий составляется именно на основе уже пережитых событий. Размышления в духе: «у меня не получится научиться рисовать,

потому что я однажды пытался и у меня вышла невнятная ерунда», берутся как раз от существования в прошлом.

В настоящем мы находимся в те моменты, когда удовлетворяем свои сиюминутные потребности: отвлечься от работы и полистать ленту новостей или посмотреть очередное бесполезное видео на YouTube – это сиюминутная потребность. Она возникает, когда мозг спонтанно подкидывает нам идею посмеяться над старыми картинками с новыми подписями, потому что когда-то они нас уже смешили и было бы неплохо посмеяться снова. Потакание подобным потребностям может сбивать с рабочего настроя, изматывать и оставлять нас без сил, но при правильном подходе погружение в настоящее является лучшим способом расслабиться и перестать муссировать мысли о прошлом или тревожиться о будущем. Я подробнее остановлюсь на феномене настоящего, когда мы будем учиться отдыхать от разработки или же наслаждаться ею.

В будущее мы попадаем, когда концентрируемся на размышлениях о грядущих днях: на предвкушении эмоций от выпуска игры, на ожидании момента, когда нас закидают деньгами и славой, в общем, на предвкушении удовлетворения той потребности, которую должна закрыть именно разработка видеоигр. Или же, напротив, концентрируемся на выдуманных провалах еще даже не начатых дел.

И вот представьте себе, что вы сидите и работаете. Вы тратите огромное количество ресурсов, чтобы сохранять внимание на объекте своего труда, но тут внезапно появляется волшебный синий джинн и предлагает абсолютно безвозмездно выполнить одно из ваших желаний.

Будете ли вы у него просить удовлетворить ваши сиюминутные желания из настоящего? Например, попросите ли его проверить ваши оповещения на смартфоне? Ответить друзьям в Телеграме? Посмотреть новое видео на YouTube? Я крайне сомневаюсь, что кто-то потратит столь редкую и ценную возможность на такую бестолковую ерунду.

Весьма очевидно, что вы попросите у джинна того, что касается вашего будущего: это может быть или новый навык, который поможет вам добиться цели, или какие-то материальные ценности. Даже несколько чемоданов денег не удовлетворят ваши сиюминутные потребности – вы будете тратить эти средства только в будущем и

обеспечивать ими именно будущее. Все ваши заветные желания улучшают вашу жизнь в перспективе, а не сию секунду.

Вы, наверное, думаете, что джиннов на свете не бывает. Но один джинн все-таки есть: это вы сами. За вас никто никаких желаний исполнять не будет. Исполнить их можете только вы.

Так почему тому джинну, который сейчас сидит и работает, вы загадываете исполнить такое бесполезное желание, как «проверить почту» или «посмотреть смешные картинки»? Почему именно этому конкретному джинну, единственному из всех существующих, вы загадываете бездельничать, в то время как вымышленным существам из сказок, обещающих выполнить любое желание, вы бы загадали что-то действительно стоящее?

В рамках понятия дисциплины каждое неправильно принятое решение выглядит как суровое противостояние на ринге. В одном углу – удушающее безделье, за которое вы будете себя ненавидеть, а в другом углу – несколько выпущенных игр, которыми вы будете гордиться. Я не попрошу у своего джинна эльфа 85 уровня в игре Lineage II, которую я так любил, но которая отняла у меня слишком много возможностей. Я попрошу у джинна три выпущенных игры.

Дисциплина заключается в исполнении тех желаний, которые пришли к вам из будущего.

Дисциплина – это ответ на вопрос, что я могу сделать в настоящем, чтобы получить желаемое в будущем?

Если вы хотите, например, прыгнуть с парашютом без инструктора, то вы включаете дисциплину, активно ходите в парашютную школу, занимаетесь там так усердно, как только можете, а потом – вуаля! – прыгаете с парашютом в гордом одиночестве. Ваше желание исполнено.

Безалаберность удовлетворяет только сиюминутные потребности, в то время как дисциплина удовлетворяет более глубокие и продуманные нужды.

Второй миф, который в наши дни свойствен образу дисциплинированного человека, заключается в том, что дисциплина включает в себя необходимость терпеть неудобства. Вставать в пять утра, мыться в холодном душе или же сидеть за компьютером по несколько часов кряду – все это можно трактовать как «неудобства».

Логическая ошибка такого утверждения заключается в том, что мы терпим неудобства не только тогда, когда пытаемся быть дисциплинированными: мы терпим неудобства все время.

В далеком детстве нас учили держать голову, чтобы та не падала назад: ее тянула вниз гравитация. Гравитация мешала нам и ходить на двух ногах, а перестать пи́саться стало вообще одним из самых сложных вызовов. Казалось бы, все эти проблемы в прошлом, но как можно оставить в прошлом гравитацию? Она всю жизнь тянет нас к земле. Прямо сейчас вашу голову что-то тянет вниз, а ваш мозг каждое мгновение контролирует состояние вашего мочевого пузыря, чтобы вы не опи́сались во время чтения. Мозг принимает и отдает такое неимоверное количество сигналов нашему телу, что пересчитать их является задачей воистину непосильной.

Если мозг перестанет выполнять свои функции в полной мере, мы упадем на землю, описаемся, опустошим кишечник, начнем задыхаться и попросту потеряем сознание в столь жалком и неподобающем виде.

Если вы носите часы, то только сто́ит на них сосредоточиться, как наличие на руке какого-то сдавливающего браслета покажется действительно неудобным. Это может касаться трусов, носков или любых других элементов одежды. Но думаете ли вы о часах, трусах и носках каждую секунду?

Мы испытываем огромное количество неудобств в любом случае. Секрет дисциплины не в том, чтобы принимать неудобства, а в том, чтобы их не чувствовать, игнорировать их. Я не игнорирую каждую секунду желание отвлечься на видеоигру или на сообщения в социальной сети – я просто не замечаю наличие этих неудобств, так же как я не ощущаю и наличие часов на моей руке или действия гравитации на мою голову.

Разумеется, я не стану игнорировать реальные призывы моего организма в случае, если ему понадобится еда или физическая разминка. Но противостояние неудобствам, которые не приведут вас к потере здоровья, вполне могут быть приравнены к контролю мочевого пузыря и должны проходить неосознанно. Со мной ничего не случится, если я сегодня не буду смотреть на котиков в Интернете.

В современной культуре жизнь в дисциплине изображена как жизнь, полная лишений. На самом же деле все обстоит с точностью до наоборот: именно современный вальяжный образ жизни с непомерным потреблением, стабильной работой и полным отсутствием увлечений обрекает людей на лишения – все желания из их будущего так и остаются просто желаниями.

18. Основы дизайна видеоигр

Неплохим способом научиться дисциплине станет… игра в видеоигры. Мы способны проводить за играми десятки часов! Но если игры станут нашей работой, то эти десятки часов мы провели, значит, не за бездельем, а за самосовершенствованием. Такое понимание позволит куда проще воспринять необходимость следующие десять часов провести за просмотром уроков и тренировкой навыков, потому что мы уже будем знать, как много времени мы готовы посвятить тому, чтобы стать лучше.

Я могу сколь угодно долго рассказывать инопланетянину о том, как выглядит кошка, но куда эффективнее будет просто показать ему это прекрасное создание и обратить его внимание на ее отличительные особенности вроде мягких лапок, умилительной мордашки и хлопковых ушек. Так и вашим лучшим учителем в освоении искусства игрового дизайна станут не книги, видеоролики или наставники, а чужие игры и личные впечатления от них.

В вашей наблюдательности кроется залог успешного освоения такой на самом деле абстрактной и изменчивой науки, как «дизайн видеоигр».

Разрабатывая независимый продукт, мы находимся в чрезвычайно выгодном положении, которое позволяет нам игнорировать «общие тренды» и формулы удержания игроков. Оставим это AAA-студиям. Нашей миссией должна стать разработка игры, которая нравится в первую очередь нам самим. Это – наш единственный путь. Если делать то, чего мы не понимаем и чем мы не умеем наслаждаться, то работа не завершится успехом: она лишь приведет нас к упадку сил и загонит в тупик.

Вам придется попрощаться с привычным процессом игры в проекты того жанра, в котором вы собрались работать. Вы не просто должны бегать по уровням и получать удовольствие. Вам нужно выяснить, что делает изучаемый вами продукт хорошим или, напротив, плохим представителем жанра. Прислушивайтесь к собственным эмоциям – какие моменты в этой игре вас раздражают, а какие, наоборот, вызывают восторг и почему.

Сосредотачивайтесь на том, что побудило вас принять то или иное решение в игре. Почему, например, вы пошли именно в «эту» сторону? Быть может, дело в том, что разработчик очень умело расставил свет и вы выбрали более освещенный путь?

Оперирование понятием «игровой цикл» поможет вам лучше понять происходящее на экране. Любая видеоигра представляет собой на самом деле повторение одних и тех же действий. У линейных платформеров игровой цикл, например, представляет собой «дойди от точки А в точку Б» – и так столько раз, сколько в игре уровней. В метроидваниях же игровой цикл заключается в следующем:

- найти босса;
- победить босса;
- приобрести новую способность;
- обнаружить области применения новой способности.

После этого мы снова находим босса и повторяем все по кругу. Игровой цикл в рогаликах сводится к изучению процедурно сгенерированной области с целью найти усиливающие вашего персонажа предметы и способности, чтобы в определенный момент убиться об очередного босса и начать все по новой.

Безусловно, я чудовищно утрирую определение игрового цикла в рамках различных жанров, но для образовательной цели нам необходимо сначала научиться мыслить общими образами, а затем уже сосредотачиваться на частностях.

> **Сформулируйте в блокнотике, в чем заключается игровой цикл игр, который вам симпатичнен и хорошо знаком. Просто записывайте действия, которые вы совершаете в этой игре, и указывайте причины, побудившие вас их выполнить. Пишите до тех пор, пока не определите, где же в этом проекте начинается новая петля.**

Внутри цикла спрятана «ключевая механика», которая обрастает «дополнительными активностями». «Ключевая механика» – это то, во что уже можно поиграть и получить удовольствие. Вспомните мой пример с Ori and the Blind Forest: даже без графики, эффектов и дополнительных механик игра увлекает.

Но стоит иметь в виду, что отнюдь не во все хорошие игры будет приятно играть в том случае, если с них срезать все детали. Я наблюдаю в разработке видеоигр два подхода: первый заключается в том, чтобы впечатлить игрока размахом, деталями и вложениями, закрыв все «дыры» вашей игры деньгами и человеческими ресурсами. Ярким примером такого подхода становятся открытые миры в современных AAA-играх, поражающие воображение игроков сумасшедшим объемом разношерстных декораций. Создание такого мира в одиночку заняло бы у разработчика целую декаду лет.

Противоположный метод заключается в «изящной экономии». В этом случае отсутствие ресурсов и возможностей выставляется так, словно автор добровольно установил себе какие-либо ограничения, обращая их в свой стиль. Самым выразительным примером служит оригинальная Silent Hill, выходившая в свое время на PS1. Ресурсов этой консоли не хватало для отрисовки открытых пространств, оттого разработчики прибегли к хитрому трюку: они замостили улицы в Silent Hill густым туманом. Как вы знаете, туман в дальнейшем стал отличительной особенностью серии.

Почти все рисунки в визуальной новелле Tiny Bunny выполнены в черно-белой палитре. Мало того, что выбор стилистики выделяет ее визуальный ряд среди других новелл и подчеркивает мрачность описываемых в ней событий, так еще и рисовать, используя лишь градации серого, гораздо проще, чем составлять сложную цветовую схему.

В Call of Duty: Black Ops – Colds War многие игроки были впечатлены тем, что к NPC можно подойти с любой стороны и начать беседу: у каждого NPC была своя анимация поворота к игроку, а сидячие NPC, к которым игрок подходил сзади, поднимались со своих мест, разворачивались и иной раз даже ворчали на игрока. Это – потрясающая деталь, но создание анимаций является очень трудоемким процессом, и соло-разработчику нерационально тратить свои ресурсы на реализацию подобной механики. Но и отнимать у игрока возможность взаимодействия с NPC только по той причине, что игрок подошел к нему сзади, тоже является весьма неуклюжим решением.

В способах выхода из сложившейся ситуации и заключается метод «изящной экономии». Если поставить NPC спиной к стене – то у

игрока просто не будет возможности подойти к нему сзади, а значит, и не возникнет необходимости дорисовывать анимацию разворота. Идея взаимодействия с NPC через закрытую дверь в Bloodborne – гениальна. Разработчики не моделировали персонажа, не занимались лицевой анимацией, не сталкивались с проблемой позиционирования при взаимодействии, они просто сделали так, что NPC не открывал нам дверь в свой дом и оставался невидимым. Изящная экономия во плоти!

Наблюдать различные способы экономии можно и в кино. Особо заметные примеры мы можем увидеть почти в любом фильме, где есть сцена перевоплощения человека в какое-нибудь страшное существо вроде оборотня. Эти сцены – очень дорогие в производстве, оттого режиссеры уже чего только ни придумали, чтобы и зрителя не разочаровать, и ресурсы сэкономить. Частенько сцена перевоплощения прерывается кадрами с изумленными лицами свидетелей сего действия; иной раз все разворачивается за спиной главного героя, и монстр находится «не в фокусе», что позволяет сделать анимацию менее детализированной; а способ показывать только тень человека, превращающего в оборотня, абсолютно великолепен – аниматор в этом случае работает только с контурами, а звуковые эффекты и воображение зрителя уже дорисуют все остальное, сохранив нужный пугающий эффект.

Подглядывайте, как экономят разработчики тех игр, в которые вы играете, и тех фильмов, которые вы смотрите. Без внимательного анализа большинство решений останется незамеченным, потому что так и должно быть – игрок не должен знать, что вы решили увести экран в затемнение на драматичной сцене лишь для того, чтобы не рисовать «одноразовые» анимации.

Любой дорогой в производстве элемент может быть использован несколько раз. Игры-рогалики являются живым примером того, сколько различных игровых ситуаций можно создать с одинаковыми игровыми элементами, но в то же время некоторые проекты вроде Devil May Cry 4 могут разочаровать игрока тем, что разработчик вынуждает его снова пробежаться по уже пройденным уровням в строго обязательном порядке.

Учитывать ограниченность ресурсов нужно еще на стадии написания сюжета. Не стоит придумывать сцены, декорации для которых будут использоваться единожды. Нет никакого смысла

показывать какого-нибудь босса сидящим в своих покоях, а потом отправлять его на встречу с игроком на специальной арене. Стройте сюжет так, чтобы использовать как можно меньше локаций и объяснить, почему босс или уже был на арене в момент нашего прибытия или же почему мы попали в его покои и решили драться там.

Мы не можем себе позволить такое расточительство, как разработчики Anthem, где детально прорисованные джунгли с горами, водопадами и десятком видов деревьев нужны лишь для того, чтобы мелькать под ногами игрока, пока он летит на реактивном ранце. Если вы добавляете в игру объект или локацию – у этого должен быть какой-либо смысл, кроме как «впечатлим игрока детализацией».

Большинство начинающих разработчиков спотыкаются об амбициозность своих проектов. В одиночку можно, разумеется, сделать что угодно – это всегда вопрос, в первую очередь, времени. На создание JRPG в открытом мире у вас может уйти больше лет, чем вам отведено на этом свете, и браться за такую игру попросту нерационально.

Качество выпускаемых вами игр будет зависеть не столько от количества потраченного на них времени, денег и сил, а сколько от используемого в них опыта. Чем скорее вы доведете игру до выпуска, тем больше опыта вы сможете вложить в следующий проект. Хватаясь за разработку игры сумасшедших масштабов, вы рискуете потоптаться на месте, потерять несколько лет жизни и выгореть. Игра, разумеется, так и не будет выпущена.

Всегда определяйтесь с ключевой механикой, вокруг которой строится ваша игра, и в первую очередь реализовывайте те вещи, без которых ваша идея не может существовать в принципе. Я замечал, что, когда я или мои коллеги рассказываем о новом, едва начатом проекте, нам обязательно задают вопрос: а будут ли в игре «пасхалки»? «Пасхалками», или «пасхальными яйцами», называют шутливые упоминания и отсылки к другим проектам. В этом нелепом вопросе меня сильнее всего мучает мысль, а какого черта я должен думать о таких мелочах в самом начале разработки? Я еще не до конца отполировал ключевую механику, о «пасхалках» я подумаю через год.

Одним из основных критериев ключевой механики является ее «понятность». Если вы зайдете в Steam, найдете там игру, которую вы любите и хорошо знаете, и начнете читать негативные отзывы, то

обнаружите парочку отзывов, которые написаны словно о другой игре. Авторы этих отзывов попросту не поняли, как играть в этот проект, и остались им недовольны. Это абсолютно нормальная ситуация, потому что игры запускают люди с абсолютно разным игровым опытом. Если вы делаете стратегию, то будьте готовы к появлению игроков, которые в стратегии никогда в жизни не играли и понятия не имеют, что такое «юнит». Ваша задача – сделать игру понятной для людей с любым игровым опытом.

Перед тем как наращивать на «ключевую механику» всю остальную игру, вам стоит убедиться в том, что эта механика понятна игрокам. Единственно верным способом такой проверки станут шоукейсы: показывайте свою игру другим игрокам на сходках разработчиков, фестивалях, просто зовите друзей к себе домой и внимательно наблюдайте за тем, как люди играют в то, что вы сделали. Шоукейс представляет собой возможность дать кому угодно попробовать поиграть в ваш проект, пока вы сами стоите рядом и общаетесь с игроком напрямую. Ни в коем случае не комментируйте и ничего не подсказывайте игрокам. Просто запоминайте, в каких местах игроки не поняли, что им надо делать, а в каких местах им было очень сложно.

Настоящая критика – это не хвалебные отзывы или остроумные замечания. Настоящая критика – это действия ваших игроков. Если игрок говорит, что игра замечательная, но уделяет ей минуту времени, он лжет. Если игрок ругает ваш проект, но не может оторваться от контроллера уже несколько часов – он лжет. Следите за действиями ваших игроков, а не за их языком.

Когда «ключевая механика» будет готова – время приступать к реализации «дополнительных активностей». Под ними я подразумеваю все, без чего игра, возможно, будет хуже, но все еще может существовать и в нее все еще можно будет играть. Чаще всего это количественный показатель: размеры уровней, численность боссов, наличие локализации на разные языки, какие-то эффекты, мини-игры и сложные элементы интерфейса. Такой порядок действий поможет вам сохранить уйму времени, потому что, поверьте, одна переделка всегда несет за собой уйму других переделок.

Однажды игрок предложил мне внести «незначительное» изменение в Fearmonium, которое позволило бы главной героине атаковать на

ходу, как это было реализовано в Catmaze. Я же шел по пути игр серии Castlevania, и атака приковывала персонажа к месту, делая бои более тактичными и вынуждая игрока с умом подходить к атаке, а не просто без остановки долбить по кнопке удара. Если бы я внес такое изменение, то мне пришлось бы править баланс, двигаясь в сторону «усложнения», потому что битвы с врагами стали бы в десятки раз проще и игрок мог бы всех убивать, просто двигаясь туда-сюда и без остановки махая молотом. Я редко использовал какие-то наводящиеся снаряды, и вся защита всегда строилась на том, чтобы двигаться. Игрок становился уязвимым, лишь когда стоял на месте. А стоять на месте он вынужден был, чтобы атаковать.

Для правки баланса мне бы пришлось пересмотреть поведения 60 монстров и 14 боссов. Честно – мне проще сделать новую игру, чем переделывать одну из ключевых механик, ибо за годы разработки вокруг этой механики выстроилось неимоверное количество «побочных элементов». Когда дом построен – поздно переделывать фундамент. А если фундамент построен плохо, то и дом в скором времени развалится, вне зависимости от того, как красиво вы облицевали стены.

> **Не стесняйтесь демонстрировать сырые сборки вашей игры. Чем раньше вы заметите свои ошибки в ходе наблюдения за игроками, тем меньше работы вам придется проделать по их исправлению.**

19. А как быстро я сделаю игру?

Неосознанно мы догадываемся, как работает мотивация, потому что начинающие разработчики обязательно поинтересуются тем, сколько времени им потребуется для достижения результата.

Я сам, когда учился рисовать, задавал художникам неуклюжий вопрос: а сколько времени ты потратил на этот рисунок? Интерес к длительности процесса приходит оттого, что мы понимаем: чтобы не сдаться, нам нужен результат. Идти по дороге образования и не наблюдать никакого прогресса – это удручающий процесс, который создает иллюзию того, что мы топчемся на месте. Вспомните бабушек и обои на рабочий стол.

История о том, что разработчик Iconoclasts потратил более десяти лет на свой проект, мотивирует только к тому, чтобы похвалить бедолагу за его железную волю, но никак не к тому, чтобы оказаться на его месте и начать делать игры. Если бы я сказал, что от момента прочтения этой книги до момента, когда ваша игра появится на виртуальных полках магазинов, пройдет десять лет, вы бы тут же перестали ее читать.

Разумеется, и 10 лет – это огромный срок даже для такой комплексной и длинной игры, как Iconoclasts. Сделать что-нибудь, во что можно поиграть полчасика, реально и за выходные на джеме, и этого уже будет достаточно, чтобы приобрести мотивацию перейти к более сложным и навороченным играм.

От момента, когда я сел осваивать движок, до момента, когда я выложил свою первую игру на джем, прошло около месяца. Нельзя сказать, что я проводил за Construct 2 все свободное время, учитывая, что у меня его тогда было в обрез: я работал воспитателем в социальном центре, учился в аспирантуре, занимался ремонтом дома и подрабатывал очень плохим художником.

Ирония заключалась в том, что перед началом джема я был уверен, что моим основным преимуществом перед более опытными разработчиками станет умение относительно сносно рисовать, но опубликованные перед началом джема правила быстро выбили меня из колеи: почти полностью запрещалось использовать собственные

рисунки, можно было брать только наборы изображений из свободного доступа.

Я потратил десять дней на то, чтобы сделать свою первую игру **Sleepy Savior**. Как видите, срок, за который вместе с подготовкой я сделал свою первую игрушку, составил всего месяц и десять дней. Но меня мало волновало время, да и вас оно волновать не должно: не нужно засекать минуты, по истечении которых вы можете с чистой совестью отбросить все начинания и сказать: «Ну все, у меня не получилось сделать игру за месяц, я бездарность, пойду-ка я займусь чем-нибудь другим».

Мы все схватываем информацию с разной скоростью и приходим в индустрию с разными навыками, некоторые из которых могут ускорить процесс обучения, а некоторые, напротив, замедлить. К последним я отношу навыки работы в программах, которые очень схожи с теми, на которые вы собрались перейти: переучиваться всегда сложнее, чем учиться, потому что помимо того, чтобы что-то запомнить, вам придется еще и что-то забыть.

Сейчас я сделаю игру, подобную Sleepy Savior, за сутки. Раньше создание анимации бега занимало у меня 10 часов. Затем – 6 часов. К своему текущему проекту я сделал анимацию бега в 16 кадров за 4 часа. Где-то всегда будет аниматор, который выдаст такой же результат за час, но какое мне до него дело? Мне важно качество результата, а не потраченное на работу время. Всегда, абсолютно на любом этапе вашего развития будут существовать специалисты, которые в два раза быстрее создадут то же самое, что сделали вы.

Создать можно все что угодно, и если у кого-то очень опытного на создание игры уйдет год, то я на данном этапе своего развития потрачу полтора года, а новичок – три года. Проецировать чужую эффективность и скорость на свои собственные умения – бессмысленно, потому что за какой бы срок кто-то ни сделал игру или ни нарисовал бы картину – вы сможете так же, но только когда наберетесь такого же опыта.

Основная деталь, которая отличает новичка от профессионала, заключается как раз в количестве затрачиваемого на выполнение задач времени. Вы можете рисовать или создавать крутые игры уже сейчас. Просто у вас на созидание уйдет больше времени, чем у людей, которые давным-давно занимаются тем же самым. И это нормально.

Но хочу отметить, что в ваших силах – ускорить этот процесс. Трюк, который я опишу, пришел из рисования, но использовать его можно в любом виде деятельности. Заключается он в том, что мастер тратит на свою работу меньше времени, чем новичок, по той причине, что драгоценные минуты не уходят на «мышиную возню», состоящую из лишней суеты и движений. Для того чтобы нарисовать линию, проходящую через весь холст, новичок сделает гораздо больше штрихов, нежели профессионал, который справится с этой задачей одним размашистым движением.

Использовать это знание можно в простом упражнении: нарисуйте несложный объект и посчитайте, какое количество движений вам потребовалось. Если вы уложились в 100 штрихов, то в следующий раз попробуйте уложиться в 80, а потом в 50. Рисунок, выполненный в 50 штрихов, будет нарисован вдвое быстрее рисунка в 100 штрихов.

Этот же фокус прекрасно используется и в программировании. Практически всегда есть способ выразить что-либо куда более лаконично, чем это делает новичок. Я смотрю на свой старый код, и он сейчас мне кажется скачками по зигзагообразной линии, нежели непринужденной прогулкой по прямой тропе.

При работе в Construct можно создать блок с «условием» для «действия». Условием может быть, например, «если нажата кнопка прыжка», а «действием» – сам прыжок персонажа. Чаще всего во время разработки блок с «условием» обрастает новыми деталями. Например, помимо условия «кнопка прыжка нажата» нужно добавить условие, что «персонаж находится на земле», «персонаж жив», «игра не стоит на паузе» и т. д. Construct подразумевает, что в один блок можно засунуть неограниченное количество условий, но по какой-то неведомой причине я очень долгое время под каждое новое условие создавал отдельный блок.

Мало того что мой «код» приобретал воистину неказистый и неудобный вид, так еще и создание каждого блока отнимало у меня несколько секунд. Дальнейшие попытки сориентироваться в монстре, которого я создал, отнимали у меня уже часы. Избавление от вредной привычки заниматься этим бездумным созданием лишних блоков сделало мою работу в разы быстрее и профессиональнее.

Новички тратят больше времени не на создание игр, а на мышиную возню и лишние движения. Противодействуйте лишним движениям в

рисовании, боритесь с излишне громоздкими решениями в коде, задумывайтесь о лаконичности в своих действиях и решениях – и вы станете намного быстрее.

Рисуйте одно и то же, совершая меньшее количество штрихов. Переписывайте громоздкие части своего кода в пользу лаконичности. Так вы научитесь работать быстрее.

20. Я – разработчик

Есть определенный тип людей, которые симпатичны мне меньше всего. Я отношу к ним тех, кто четко понимает, какая конкретно цель должна быть достигнута для удовлетворения их потребностей, но при этом не прилагает никаких видимых усилий для ее достижения.

Шесть лет назад мой знакомый с пылким во-одушевлением делился со мной планами по запуску одного весьма перспективного онлайн-сервиса. Сам он работал менеджером среднего звена. Навыки веб-программирования он приобретал в свободное время и никуда особо не спешил. Пять лет назад он все еще хотел реализовать все те же идеи. И четыре года назад. И три. Да даже в этом году при встрече с ним я точно знаю, что буду слушать то, как он с трепетным воодушевлением рассказывает о перспективном будущем своего сервиса. Такие же слова я услышу и в следующем году, и через два года, и через пять лет.

Почему так происходит? У человека вроде есть неудовлетворенные потребности, уже сформирована приблизительная программа действий, накопился массивный запас мотивации – в чем же тут пробел? Чего ему не хватает, чтобы изменить свою жизнь?

Мы не можем поменять свою жизнь, не изменив себя. Если в моей шкуре внезапно окажется другой человек, то мои дела пойдут совсем по иному пути, потому что он будет иначе мыслить.

Восемь лет назад, когда я еще работал воспитателем в центре временного содержания несовершеннолетних, я был не способен принимать те решения, которые принимаю сейчас. И дело не только в моих нынешних навыках, знаниях и опыте, дело в том, как теперь работает мое мышление и какие убеждения я приобрел. Если бы мое мышление осталось прежним, то я все еще идентифицировал себя как «воспитатель», но сейчас я – разработчик.

У нашего мозга есть очень неприятная особенность, и заключается она в сопротивлении изменениям. В условиях перемен ему труднее просчитать риски, оттого он предпочтет этих перемен избегать и создаст нам образ «самого себя», за который мы будем цепляться изо

всех сил, даже если это разрушает нашу жизнь и перечеркивает наши мечты.

Мозг будет изо всех сил пытаться оставить нас такими, какие мы есть сейчас, особенно если базовые потребности у нас удовлетворены. Монолог нашего мозга может звучать как-то так: «Ты сыт?.. Да плевать, что в ресторан тебе не сходить, мне до этого дела нет. Эти вопросы – к твоему «эго». Каши овсяной же ты купить сможешь, верно? Ну вот и все, значит, сыт. А дикий зверь за тобой не гонится? Нет? Ну так сиди и не рыпайся. Разработчиком, видите ли, он захотел стать. Ты – бухгалтер. Нам ничего не угрожает, пока ты бухгалтер, а что там с тобой будет, когда ты станешь, видите ли, разработчиком, я понятия не имею. Я не хочу лезть в это непонятное болото, вдруг там даже каши нет. В общем, шиш тебе, а не мотивация стать разработчиком. Сиди и таблички свои заполняй».

Вы можете сколь угодно долго говорить: «Я – разработчик», но это не поможет надломить ваш упертый мозг в его твердом убеждении, что куда выгоднее сохранять свое нынешнее «я», чем внезапно приобретать новую роль.

Множество наших неверных решений исходит из того, что мы боимся навредить своей идентичности. Наш имидж и образ отчасти определяют наше поведение. Вы знаете себя как бухгалтера, и взбалмошное проявление креативности может разрушить образ рассудительного и спокойного человека. Причем не только в глазах других людей – в ваших собственных тоже.

Проводилось множество экспериментов и исследований, доказывающих, что, выбрав что-то одно, человек с бóльшей вероятностью будет принимать все последующие решения исходя из того направления, с которым он определился в самом начале. Тот, кто когда-то решал, хочет он стать менеджером или художником, будет неосознанно пытаться увести свою карьеру как можно дальше от той специальности, которую он отсёк.

Кто-то рекомендует «убить свое “я”», чтобы познать себя заново как абсолютно иное существо, но такой подход отдает эзотеризмом, а сам совет является востребованным только потому, что он звучит по-бунтарски круто и чрезмерно лаконично.

Дела обстоят немного иначе. Наше как таковое «я» существует одновременно в трех вариациях: есть «реальное я», «идеальное я» и

«социальное я».

«Реальное я» – это то, кем мы себя ощущаем в данный момент. «Идеальное я» – это то, кем мы хотим себя ощущать. Я упущу подробный рассказ только о «социальном я», которое представляет собой то, как мы хотим восприниматься другими людьми. Современные технологии практически объединили эту форму самовосприятия с «идеальным я»: наша социальность – это Интернет, а там мы можем регулировать свой образ, подгоняя его под идеальное представление о себе.

Мой вышеупомянутый знакомый, работающий менеджером и грезящий о собственном онлайн-сервисе, чертовски крепко вцепился за свое «реальное я». Его мозг знает, что этот человек пытается превратиться в свое «идеальное я», но прогнозировать неясные риски нового «я» – это дело, которого мозг попытается избежать, всячески препятствуя переменам и в характере, и в манере поведения моего несчастного товарища.

Именно этот конформизм и желание сохранить свое «я» определяют поведение людей, совершающих новые выборы в «поддержку» своих старых решений. «Не создавать онлайн-сервис» – это тоже выбор, который мой приятель принимает каждый день.

Еще одна любопытная деталь заключается в стремлении «реального я» к комфорту и покою – к ситуации, в которой мозгу легко просчитывать риски. Мы можем быть недовольны собой, а между нашим «идеальным я» и «реальным я» может пролегать огромная пропасть из непокоренных вершин и неприобретенных навыков, но мозг всегда будет цепляться именно за текущую форму самовосприятия и всячески препятствовать достижению «идеального я».

Подумайте о том, кем вы бы хотели себя видеть и почему. Если вы читаете эту книгу, то явно разработчиком видеоигр. Образ творца-одиночки может привлечь по разным причинам: кто-то представит себя на крутых конференциях; кто-то будет фантазировать о том, как он читает тонны положительных отзывов на свою игру; кто-то хочет оказаться на интервью на определенном YouTube-канале.

Образ «идеального я» у всех разный, но чаще всего он сопряжен с адреналином, приключениями и рисками. Вы не мечтаете лежать дома за приставкой и чесать животик, вам хочется свершений, признания и

самовыражения, что сопряжено с рисками быть непонятым, униженным и оскорбленным.

Сопротивляясь переменам, мозг может подхватить болезненную веру в то, что начатое дело нужно бросить, ибо все самые страшные сценарии непременно сбудутся. У нас в голове загноится язвой иррациональная уверенность. что «игра не найдет своего игрока», «она никому не понравится», «у нас ничего не получится» и т. д.

Вооружившись гнусными мыслишками, мозг будет подло подставлять подножку каждый раз, когда мы попытаемся изменить свое «я». Но теперь мы знаем механизм, провоцирующий появление таких катастрофичных и унылых убеждений: наш мозг боится за нашу идентичность. Вся эта ерунда, которой он нас пугает, – это лишь невнимательно просчитанные им риски. Самые худшие сценарии мозг использует для того, чтобы мы никогда не менялись.

Чтобы утверждение «мою игру никто не увидит» стало истиной, нужно сначала в этом убедиться, а чтобы в этом убедиться – нужно сделать игру и посмотреть, увидит ли ее кто-то или нет. В противном случае это не то утверждение, исходя из которого стоит строить свои планы на жизнь. Это – просто суетливый домысел.

> **Вам нужно признать, что новая профессия отчасти сделает вас новым человеком с новыми увлечениями и новым кругом знакомств. Вам не избежать перемен, и это… прекрасно. Если бы вы не хотели перемен, вы бы не стали думать о том, чтобы делать игры.**

Перед тем как начать неистовый марш к своему «идеальному я», это «идеальное я» нужно сначала сформировать. В этом процессе нам может помочь старая схема, по которой мозг ищет в воспоминаниях способы удовлетворения наших потребностей.

Когда я посмотрел **Indie Game: The Movie** меня сильнее всего впечатлил образ Эдмунда Макмиллена – одного из разработчиков великолепной игры **Super Meat Boy**. Я достаточно быстро узнал, что в кино его представили весьма однобоко и он отыгрывал там роль уверенного в себе жизнерадостного толстячка, но в абстрактности этой роли и крылся ее шарм. Взяв за основу столь выразительный образ я и сформировал свое «идеальное я». В начале своего пути я стал

рассуждать над проблемами в духе «а как бы Эдмунд решил бы этот вопрос?» Конечно, находясь на прежнем месте работы, я не мог знать, как Эдмунд заполнил бы документы на подростка, которого привезли ко мне ночью на отделение в сопровождении толпы полицейских. Но подражание непринужденности и жизнерадостности его персонажа помогало мне принимать решения с куда большей уверенностью и легкостью. Я потихоньку сближал свое «реальное я» с «идеальным я».

В процессе подражания не стоит заходить далеко и создавать себе кумиров. Напротив, образ должен быть максимально абстрактен, чтобы давать вам больше свободы в выборе действий в новой роли. Хоть Эдмунд и стал для меня идейным вдохновителем, но он мне знаком лишь по роли в вышеупомянутом великолепном фильме и по двум интервью, что довелось прочитать. Углубляться в биографию и характер этого человека я не стал – не хочу разочароваться в том, каков Эдмунд на самом деле, и не собираюсь сделать рамки своего «идеального я» катастрофично узкими.

Мне немного жаль, что настоящая «я-концепция» не подразумевает существования какого-нибудь «отвратительного я», потому что держать в голове образ существа, на которое мы крайне не хотим быть похожими, является занятием весьма полезным. Я чрезвычайно боюсь казаться нытиком, и мое «отвратительное я» представляет собой беспомощного и плаксивого персонажа. Как только мое поведение и мои решения начинают совпадать с поведением и решениями, которые принял бы этот капризный дурачок, я одергиваю себя и пытаюсь взять снова собраться.

Ситуацию еще усугубляет наш конформизм, который является иерархическим страхом – страхом потерять свою стаю. Пока у вас есть коллектив, а в коллективе вы занимаете не самый низкий статус, то мозг будет цепляться за возможность сохранить текущее положение дел, избегая риска остаться без стаи. В этом контексте мой совет посещать сходки разработчиков должен прозвучать еще более убедительно, потому что вам необходимо показать вашему мозгу, что вот же она, новая стая.

Если хочется перемен, то стоит убедить свой мозг, что оставаться таким, какой вы есть сейчас, попросту опасно и чревато тем, что рано или поздно вы потеряете возможность удовлетворять ваши растущие потребности. Вам придется перестать воспринимать свою нынешнюю

роль как что-то положительное, потому что вам нужен стимул для перемен. Я думаю, он у вас уже есть – просто ещё раз вспомните, зачем вы собрались делать игры в одиночку и зачем читаете эту книгу.

Только не нужно думать, что я призываю начать ненавидеть свою нынешнюю жизнь. Это было бы безусловно очень эффективным способом найти достаточное количество мотивации для перемен, но провоцировать возможную депрессию и чрезмерно завышенную тревожность мне не хочется. Я призываю сконцентрироваться на том, что именно вы хотите поменять в своей жизни, раз решили сменить вектор своего развития. Вряд ли вами движет простое любопытство.

21. Я – один, и это хорошо

То, что один программист сделает за один день, два программиста сделают за два дня.

IT-мудрость

Причины, по которым мы выбрали делать игры именно в одиночку, у всех разные, но вот трудности и прелести, с которыми мы сталкиваемся на пути одиноких воинов, у многих из нас будут схожими. Поговорим сперва о прелестях.

У меня был опыт работы в команде, которая благополучно развалилась; я консультирую различные студии разработчиков перед выпуском их игр и наблюдаю все внутренние процессы, ошибки и трудности взаимодействия. Я насмотрелся на тот, другой берег, где люди толпятся плотными рядами и работают вместе. Я вполне осмысленно остаюсь Робинзоном на собственном острове с пальмами, пляжем и мячиком Уилсоном.

После полученного опыта я твердо укоренился в своем личном убеждении: самое трудное в разработке видеоигр в одиночку сводится к тому, чтобы перестать наконец-то восторгаться тем, насколько же это, черт возьми, классно – делать игры одному!

При работе в одиночку вам предстоит заниматься только теми проектами, которые вас вдохновляют. Вы вольны делать игру в любом жанре, который вам по душе. Крайне мала вероятность того, что так же сильно, как своя собственная идея, вас зажжет идея чужого проекта. Мне сложно представить какой должна быть игра, чтобы я занялся ее разработкой с таким же восторгом, с каким я занимаюсь своими собственными проектами.

Второй плюс заключается в том, что еще на стадии формирования идеи мы ставим перед собой те задачи, которые хорошо понимаем. Никто не сможет мне объяснить, что нужно что-то нарисовать или написать так же четко и понятно, как я сам. Каким бы красноречием не обладал мой коллега, передать свое видение таким, каким оно

предстало у него в голове, невозможно. Он и сам едва ли осознает до конца то, о чем меня просит.

При разработке, например, платформера, очень важно проработать механику прыжка. И если я в команде занимаю должность программиста, то задание «сделай прыжок как в «такой-то игре»» от игрового дизайнера является весьма адекватным и логичным. Но вдруг мне не нравится прыжок в упомянутой игре? Смогу ли я повторить чужую механику, если она мне не симпатична и кажется сломанной? Сортов вещей, которые любому из нас неприятны, очень и очень много, но разницу между ними уловить крайне затруднительно, а выделить особенности чего-то субъективно хорошего – куда проще. Может быть, мне и удастся повторить чужую механику, но, если она мне не нравится, я, перебирая сотни вариантов реализации, не смогу идентифицировать тот самый, который нужен нашему игровому дизайнеру.

Формулируя задачи для самого себя, я куда четче вижу, каким должен быть результат.

Третий плюс работы в одиночку заключается в том, что еще на стадии планирования проекта мы сможем учесть свои сильные и слабые стороны. В моей игре Catmaze, например, всего два мужских персонажа. Это связано с тем, что мне неинтересно рисовать мужчин и, если честно, я не особо-то хорошо справляюсь с этой работой. Еще при написании сценария я учитывал отсутствие навыка справляться с дизайном героев мужского пола и тем самым определил пол ключевых действующих лиц.

Безусловно, хороший лидер будет считать, что он знаком со слабостями и достоинствами своих подопечных и пытаться учесть их, но, к сожалению, познание своих сильных и слабых сторон – это бесконечный процесс полный неожиданных открытий для самого творца. Разрабатывая все тот же Catmaze, я и подумать не мог, насколько сильно мне понравится рисовать монстров, и в процессе создания одного из них я прибегнул к четвертому плюсу работы в одиночку.

Соло-проект является куда более гибким. Когда я, сидя за компьютером в час ночи, внезапно решил сделать из рядового монстра ключевого героя повествования и вплести его в сюжет, мне не требовалось проводить долгие беседы со своими коллегами о

целесообразности реализовывать столь внезапную идею. Я избежал утомительных бесед с игровым дизайнером; мне не пришлось формировать техническое задание для художника, от которого потребовался бы увеличенный портрет этого монстра; не возникла необходимость вынуждать дизайнера уровней придумывать, где этот персонаж будет обитать. В час ночи я придумал персонажа и историю, связанную с ним, а к семи утра уже закончил вплетение этого героя в игру. Мои коллеги бы все еще спали.

Пятый плюс заключается в том, что я, будучи и художником по локациям, никогда не буду вымучивать не вдохновляющие меня фоны или, будучи дизайнером персонажей, никогда не буду работать в неприятном для меня стиле. Лично мне больше всего нравится рисовать природу, поэтому я не являюсь большим фанатом урбанистических локаций вроде каких-нибудь складов или заводов. Тогда стоит мне переносить действие своей собственной игры на завод? Конечно, нет – ведь иначе мне придется его рисовать, а я не хочу рисовать завод.

Все локации, все герои и абсолютно все события моих игр мне глубоко симпатичны. Если же я окажусь на проекте, где действия разворачиваются, например, в индустриальной среде, а рисовать мне придется различного рода современные механизмы, то, конечно, не буду творить с таким же воодушевлением, с которым творю сейчас. А что самое ужасное, я не смогу отличить завод, который хорошо нарисован, от завода, который плохо нарисован, потому что мне не будет нравиться нарисованное в любом случае. Угадайте, как это в итоге скажется на качестве проекта?

Если я почувствую, что мне не нравится работать над каким-либо элементом, то вполне могу убрать его из своей игры.

Шестой плюс заключается в том, что удается сэкономить кучу времени, не участвуя в бесконечных видеоконференциях между членами команды. Вам не придется озвучивать свой «план на неделю», а потом отчитываться, почему вы его не выполнили. Вам не придется отстаивать свои идеи, расходуя силы и нервы на придумывание аргументов в пользу их реализации. Что сами придумали, то в игру и пойдет. Также вам не придется доказывать, почему, например, катающиеся на аэродинамических скейтбордах устрицы – плохая идея

для игры про динозавров. А ведь иной раз коллеги и не такой абсурд предложат!

Вам не придется продумывать уникальный подход к каждому из своих партнеров. Не нужно будет подбирать слова, которые не покажутся им обидными, когда вы будете критиковать их работу. Вам не придется изобретать и запоминать методы убеждения коллег в своей правоте. Выстраивание межличностных отношений в команде – это тяжелый труд. И самое неприятное заключается в том, что не все это понимают. Пока вы пытаетесь придумать подход к товарищу, ему до таких тонкостей совместной работы не будет никакого дела: он продолжит изъясняться так, что его коллеги всегда будут неправильно трактовать его слова, а кто-то – даже обижаться, потому что растолкует нечто, сказанное им, как хамство.

Все, что я перечислил, отнимает невероятно много времени. Лично у меня сформулировано весьма суровое разграничение между временем, которое я считаю продуктивно проведенным, и временем, которое я воспринимаю потраченным впустую. Реализация идей – это продуктивно проведенное время. Доказывать кому-то, какие идеи хороши, а какие нет, балансируя в чате на грани конструктивной критики и бесполезного обмена сообщениями с шутками и мемами, – это непродуктивно проведенное время, являющееся бессмысленной суетой.

Никто не будет спорить с мыслью, что получать удовольствие от работы – это очень важно, ведь так? Тяжело возвращаться к делу, которое не приносит радости, и получение радости от разработки – это ключевое составляющее в профилактике выгорания.

Доделанная до конца работа (или хотя бы видимая часть работы) – это колоссальный источник радости. Здесь дела обстоят точно так же, как и с обучением: видимый результат дает прилив сил. Когда я работаю один и, например, доделываю какую-то механику, я любуюсь тем, как она работает, и получаю удовольствие в тот же миг, как работа была завершена. Лицезрение нового элемента в своей игре является моей наградой. Чертовски приятно наблюдать свое детище в действии. Это дает мне сил приступить к реализации новых идей и веру в то, что все у меня получится.

В работе же с командой я никогда не мог быть уверен, что добавленная мной в игру механика или персонаж не будут

подвергнуты изменениям. Перед тем, как моя работа отмечалась как «сделанная», ее должны были заверить и обсудить члены команды, что, разумеется, происходило не в ту же секунду, как я добавлял что-то новое в проект. Всем предварительно нужно было ознакомиться с моими нововведениями, обдумать их, затем созвониться, обсудить каждый аспект и, возможно, что-то подправить. Затем составить техническое задание программисту или художнику и терпеливо ждать, пока то, что сделал я, будет синтезировано с результатом его работы и появится наконец-то в игре.

Прекрасное чувство «я доделал» появлялось через огромный промежуток времени после того, как я ставил точку в своей работе. Далеко не факт, что всплеск гормонов радости в тот момент мне был так же необходим, как когда я закончил одно дело и готов был приступить к другому. Более того, совсем не обязательно, что всплеск гормонов радости вообще произойдет: может быть, полное одобрение вовсе не будет восприниматься как награда, потому что между утомительной чередой обсуждений и совещаний я забыл, что конкретно я делал, и уже успел погрузиться в другие задачи.

Предпоследний плюс, который я хочу отметить, заключается в том, что все ошибки на собственном проекте – это наши и только наши ошибки. Я не делаю идеальных игр, которые способны угодить всем и собрать десятки тысяч положительных отзывов. Мои игры не стали суперхитами, но каждый новый мой проект становится лучше предыдущего, ведь можно сколько угодно повторять: «Учитесь на чужих ошибках», но обучение на чужих ошибках далеко не так эффективно, как обучение на своих собственных.

Я не делегирую свои обязанности другим людям и ни с кем не разделяю ответственность за свои косяки. Любой негативный отзыв – это не вина программиста, игрового дизайнера или художника. Это – моя вина и моя ответственность.

С выходом новой игры, разработанной в одиночку, я получаю куда больше знаний и опыта, чем если бы я был на проекте просто, например, художником. Художнику нет дела до отзывов в духе «у игры неотзывчивое управление и плохой игровой дизайн». Художник может свалить провал проекта на своих коллег, которые занимались этим самым управлением и дизайном, и с чистой совестью считать свою работу выполненной на ура.

Я же наберусь опыта и продолжу развиваться «вширь», совершенствуясь в разных направлениях, а более того – я еще и заработаю больше.

Последний плюс является самым очевидным и касается вышеупомянутого заработка. Моя первая игра – Reflection of Mine – не стала суперхитом и продемонстрировала весьма средние продажи. Но доход с нее позволил мне уволиться с моей предыдущей работы и начать посвящать все свое время разработке игр. А представляете, что было, если бы мне пришлось делить этот доход с кем-либо еще?

Я бы так и оставался воспитателем.

Когда вы один, то сами можете подстраивать содержание вашего проекта под ваши таланты, предпочтения и личностные особенности. Вам будет проще придать игре целостности, потому что, например, сюжет будет учитывать ваши навыки рисования, а внедряемые в игру механики – ваши навыки программирования.

22. Я – один, и это плохо

В этом мире не существует правильного или неправильного. Только черная бездна человеческого сердца определяет хорошее и плохое.

Pascal's Wager, про́клятый мечник Геральд

Почти не существует ничего объективно хорошего или объективно плохого. Даже у самого налаженного процесса всегда есть свои недостатки, с которыми приходится смириться.

Этот процесс расходует силы, а вещи, которые вы воспринимаете как что-то, что нужно «терпеть», полностью зависят от ваших личностных особенностей. Кому-то в работе придется «терпеть» необходимость принимать решения самостоятельно, а кто-то может насладиться выяснением отношений с коллегами при работе в команде.

Работая в одиночестве, я не трачу силы на «терпение» – меня все устраивает. Потому мне пришлось для текущей главы задать на своей публичной странице вопрос: «В чем, на ваш взгляд, заключается главная трудность при работе над игрой в одиночку?» Полученные ответы я вынес на эти страницы.

Первая, самая весомая, по мнению подписчиков, проблема уместилась в чрезвычайно лаконичное и чуждое мне утверждение: через длительное время разработки видеоигры в одиночку приходит ощущение, что ты делаешь никому не нужную ерунду.

Я никогда прежде не задумывался о подобном отношении к собственной деятельности, потому что разделение труда на «полезное» и «бесполезное» является таким же абсурдным, как деление мира на черное и белое. У всего есть оттенки.

К объективно полезной работе можно отнести только труд врачей и спасателей, а без деятельности всякого рода актеров, спортсменов, музыкантов, художников, журналистов, писателей и разработчиков

видеоигр, казалось бы, вполне можно обойтись. Иными словами, если взять за истину утверждение, что «разработка игр – это ненужная ерунда», то единственно верный путь должен всех нас привести в медицинское училище.

Когда я был воспитателем, то занимался отчасти объективно полезной деятельностью: я проводил время с детьми, оставшимися без опеки своих родных, играл с ними, учил их чему-то новому, читал им книги и помогал делать домашние задания. Весьма внушительную часть времени я, правда, просиживал штаны на планерках с коллегами, заполнял абсурдное количество документации и переставлял местами абсолютно одинаковые огнетушители, потому что на них были номера и по плану отделения они должны были стоять каждый на своем месте.

Если следовать утверждению о бесполезности маленьких видеоигр, то мою деятельность по разработке можно было бы отнести к никчемному «перетаскиванию огнетушителей», а не к социально значимому «воспитанию детей». Как же у меня получается, после того как я приносил объективную пользу обществу, со спокойной совестью просиживать штаны за разработкой ненужных игр?

А получается потому, что я помню восторг детей, когда я показывал им свои игры. Я не забуду, как всем отделением ребята собирались у компьютера и пытались решить головоломку, в создании которой принимал участие один из них. Я видел, с каким воодушевлением после моих рассказов о разработке они брались за новые для себя задачи: в них загоралась вера, что они могут сделать самостоятельно абсолютно что угодно. Самые бойкие хулиганы успокаивались, ломая голову над моей Reflection of Mine, потому что прохождение некоторых уровней в их компании становилось очень статусным достижением.

Сейчас, когда я читаю некоторые особо трогательные отзывы на свои игры, я могу и прослезиться. Однажды мне пришло письмо с подробным рассказом о том, как Fearmonium поспособствовал выходу игрока из затяжной депрессии. Игра стала светлым лучом в тяжелый период его жизни, а сюжет позволил поверить в свои силы: мысли, возникшие у него при игре в Fearmonium, помогли обрисовать тот путь, что пролегает от его нынешнего состояния к счастливой жизни.

Безусловно, я не получаю такие письма каждый день – мои игры не столь популярны, чтобы дотянуться до всех игроков, которым они

могут помочь. Но одно такое письмо в год уже дает достаточно причин, чтобы отнести видеоигры к «полезной» деятельности и забыть нелепое утверждение, что «делая игру, я занимаюсь ерундой».

Делая игры, мы создаем нечто, что поможет людям стать чуточку счастливее. Разве это недостаточно важное дело?

Второй недостаток работы в одиночку, по мнению подписчиков, заключается в том, что у одиночек куда ниже потолок качества проекта: один человек не может быть мастером абсолютно всех дел и в каких-то аспектах игры обязательно появятся просадки.

Подобное утверждение звучит убедительно. Действительно, каким образом у программиста получится нарисовать хорошие текстуры, а у художника – написать отличный сценарий?

Но считаете ли вы, что все сценарии в ААА-играх являются эталоном непревзойденной работы? Над ними же трудились отдельные сценаристы, которые ничем, кроме построения сюжета, не занимаются. Сценарии к высокобюджетным фильмам пишут не операторы или мастера по декорациям, а специально нанятые люди, которые занимаются только одной-единственной деятельностью. Но только вот иногда они выдают такую лютую чушь, что при просмотре стыдно становится даже зрителю.

Если бы секрет качества заключался бы в том, что каждым аспектом продукта должен заниматься отдельный человек, тогда бы ребята, работающие в команде, создавали бы только шедевры. На практике все отнюдь не так, а «мастеров дела» не существует вовсе: насколько хорошо нужно, например, уметь рисовать, чтобы решить, что ты уже годишься на роль игрового художника? Где эта грань? Если действительно талантливого художника спросить, достиг ли он совершенства в своем деле, то он ответит отрицательно. Нам всем абсолютно на любом уровне знаний и навыков всегда есть к чему стремиться и куда развиваться.

И всегда будет кто-то лучше, чем мы.

Освоение какого-либо навыка – это бесконечный и очень увлекательный путь, и на его обочине не установлено знака: «С текущего момента и далее ты достаточно хорош, чтобы делать игры». Идеально отполированные аспекты видеоигры в сумме могут дать абсолютно невнятный и стерильный результат, в то время как схематичная простота некоторых элементов способна придать проекту

уникальный шарм. Графику в Undertale едва ли можно назвать восхитительной, но вся игра создана с учетом возможностей Тоби Фокса, и упор в ней делался совсем на другие вещи. При этом есть множество замечательных игр, которые лишены хитрых сюжетов, продуманных героев и сложных диалогов – игроки могут полюбить такие игры или за атмосферу, или за игровой процесс. Существует и пласт игр, основанный на очень простых игровых механиках, реализация которых осуществляется крайне несложными манипуляциями в движках – взять хотя бы To The Moon или весь жанр визуальных новелл. Тем не менее аудитория таких проектов воистину огромна.

Разработчику-одиночке не нужно осваивать все аспекты разработки на заоблачном уровне. Ему достаточно знать все о своих слабых и сильных сторонах и адаптировать свои идеи под свои умения. Если в вашей игре нет сюжета, то и ваше нежелание развиваться как сценариста никоим образом не отразится на качестве работы. А вот если вы работаете в команде и к вам устроился сценарист, которому просто нужда еда, тогда в игре появится дурацкий сценарий и придурковатые персонажи.

Перечисление успешных проектов, которые были разработаны в одиночку, иной раз прерывается вальяжным взмахом руки и небрежно брошенной фразой: «Это все просто ошибка выжившего». Действительно, возможно ли посчитать, сколько игр, сделанных соло, остались незамеченными и убыточными? Насколько правильно приводить в пример только успешные продукты?

Контраргументом здесь будет встречный вопрос о том, «а сколько игр, разработанных в **командах**, провалилось?» Я не думаю, что существует подобная статистика, но весьма очевидно, что как работа в одиночку, так и работа в команде не гарантируют успеха игры. Наличие или отсутствие дополнительных людей, работающих над проектом, никоим образом не является залогом его успешности.

Кто-то приводил аналогию со строительством, сравнивая создание видеоигры с постройкой дома. Утверждение звучало так: «Одному можно соорудить будку или небольшой домик, но если хочется отправиться в космос, то в одиночку собрать космический корабль у вас не выйдет».

Очень хорошо, что существуют еще люди, которые строят будки – иначе космических кораблей было бы куда больше, чем космонавтов. Не у всех разработчиков есть желание создавать ААА-игру с открытым миром, вышками и реалистичной графикой. Так же как не у всех игроков есть желание в это играть: являясь редактором публичной страницы «ИНДИ ИГРЫ» в VK, я вижу, какое огромное количество людей, выбирая между очередным ААА-блокбастером и непонятной игрой от независимых разработчиков, отдают предпочтение непонятной игре от независимых разработчиков.

Не обязательно набивать активностями многие километры игрового пространства для того, чтобы самовыразиться, – кому-то хватает для этого линейного приключения длиной в пять часов.

Задумайтесь, для каких целей вы создаете игру, что именно вы хотите сказать? Нужен ли вам для этого объем игры ААА-уровня с многомиллионным бюджетом? Быть может, вы искренне желаете просто приютить щенка, и все, что вам нужно, это будка, а не космический корабль?

Множество критики проповедуемого мной подхода «делать игры одному» заключалось в трудности оценки объема работ, что невольно приводит к ситуации, когда проект превращается в изнурительный, демотивирующий долгострой. К этой же категории замечаний я бы отнес утверждение, что при работе в одиночку часто сталкиваешься с «сюрпризами» из-за недостатка опыта в той или иной узкоспециализированной сфере, а также из-за вероятности застрять в «переделках».

Я объединил эту критику в один блок, потому что на все эти проблемы есть один ответ: игровые джемы! Делая небольшие игры, вы и научитесь оценивать объем работ, и обретете навык предвидеть «сюрпризы», а также избавитесь от навязчивой идеи переделывать проект каждый раз с нуля, потому что у вас будет знание о том, что финализировать игру и взяться за разработку новой – гораздо проще и эффективнее, чем переделывать старый проект.

Предпоследнее, малопонятное мне замечание, попало на страницы этой книги только ради весьма забавного примера из моей практики. Кто-то в комментариях утверждал, что разработчик, делающий игру в одиночку, автоматически загоняет и проект, и свой потенциал в рамки.

Мой пример заключается в том, что, когда я искал свою первую работу художником, мое портфолио состояло в основном из нарисованных женщин и парочки фонов – со всем остальным я справлялся плохо. Но один из моих первых заказов требовал от меня того, чтобы я нарисовал более восьмидесяти животных для детской мобильной игры. Умел ли я рисовать животных? Нет. Были ли мне интересные детские игры? Нет. Раскрыл ли я свой потенциал? Судите сами: вот одна из зверюшек, которых я из себя с огромным усилием воли-таки выстрадал*(рис. 12)*.

Рис. 12. ***Гиена, нарисованная мной для детской игры за 50 рублей, 2013 год***

Иными словами, именно работа в одиночку позволяет мне раскрыть свой потенциал, в то время как работа на сторонних проектах вынуждала меня рисовать абсолютно безыдейные и неинтересные мне вещи. Однажды к солнечной игре с чрезвычайно милыми персонажами и котиками меня попросили добавить разлетающуюся во все стороны кровь. Интересно, как принятие этой просьбы помогло бы мне раскрыть свой потенциал и выйти за рамки? Подобная просьба от

коллеги скорее перечеркивала сформированный мной графический стиль и идею, которую я вкладывал в свою работу. Ему нужна была не сама кровь: он хотел видеть бóльший отклик от столкновения персонажа с препятствием и зачем-то сам стал придумывать, каким образом этого отклика можно добиться. Ему стоило осознать проблему, которую он планировал решить с помощью крови, и попросить художника решить вопрос о недостаточной «сочности» анимации столкновения. Искр и блесток вполне хватило, чтобы урегулировать вопрос без кровопролития.

Последний блок замечаний к разработке игр в одиночку выводит нас к новой главе. Чрезвычайно часто люди жалуются, что в командах им проще генерировать идеи, в то время как в одиночестве остро ощущается нехватка воображения и формируется ограниченный взгляд на игру. Эта проблема – прямое следствие непонимания некоторых аспектов работы нашего мозга.

Работая в одиночку, вы встретитесь с определенными сложностями, и только вам решать, идентифицируете ли вы их как непреодолимые преграды, или же, не глядя, перешагнете через невозможность сделать ААА-игру в открытом мире, отдав свое предпочтение созданию короткого и эмоционального приключения.

23. Там, где живет фантазия

Обучение созданию видеоигр строится на повторении уже придуманных до вас механик. Но, чтобы созидать, а не просто накапливать знания, вам нужны идеи в том направлении, в котором вы собрались двигаться: будете ли вы делать двухмерную игру или предпочитаете три измерения? В каком жанре вы собрались работать?

Когда дело доходит до практики (до вашего первого игрового джема), размытые образы еще несозданных игр должны обрасти конкретными деталями: вам придется продумывать сотни мелочей, изобретать интерфейсы, объяснять игроку, почему ваш герой орудует именно мечом, а не дробовиком, продумывать отклик от действий и поведение монстров в маловероятных ситуациях.

Если же никаких представлений о собственном проекте у вас так и не появилось, то для какой, на ваш взгляд, деятельности будет вырабатываться психическая энергия? Разумеется, нас может осветить краткий миг озарения, когда яркая мысль якобы сама пришла голову и подтолкнула нас на путь разработчика игр. Но нелепо полагать, что идея эта явилась без приглашения: моменту озарения на самом деле способствовало множество факторов.

В наших силах этими факторами управлять и провоцировать появление идей.

Отвлекитесь немного от чтения и представьте себе будку. Лаконичную деревянную будку с треугольной крышей, звонкой цепью и металлической миской в радиусе пары метров. Стоит это чудо на зеленой лужайке около загородного дома, а рядом лежит небольшая косточка. После того как этот образ прорисовался у вас в сознании в мельчайших деталях, закройте глаза и представьте себе трех животных.

Представили? Я почти уверен, что среди них был хотя бы один пес. Если нет, то или нечто другое ассоциируется у вас с будками, или же за то короткое время, что прошло с начала чтения абзаца, что-то иное вас отвлекло и по ассоциативной цепочке вывело ваш разум к образу другого животного. Если же все пошло по плану, то это книга заложила вам в голову образ будки, и отталкиваясь от него, ваш мозг

сгенерировал образ пса. Никаких собак я здесь не упоминал – вы сами придумали собаку.

Чаще всего такая игра с воображением воспринимается как некий психологический фокус или притча в духе «не думай о белой обезьяне», но важно понимать, что именно в этом «фокусе» и кроется один из ключей к нашему воображению и к механизму генерации новых идей.

Для полного понимания процесса созидания, нырнем в глубокое прошлое. В средневековой Германии некогда жил один барон, о брутальной суровости которого слагали легенды: в одной из битв ему отсекли руку, но вместо того чтобы принять роль страдающего калеки, барон продолжил участвовать в битвах и установил себе железный механический протез, нагонявший на врагов еще больше ужаса, нежели настоящая рука.

Несколькими веками позже всем известный Иоганн Вольфганг Гёте сделал этого барона героем своих произведений, наделяя его еще более выразительным характером жестокого и отчаянного воителя с железной рукой.

Уже на нашем веку, но совсем на другой стороне света, появляется гениальная манга Berserk, повествующая о приключениях брутального воина с железной рукой. Носит этот воин имя Гатс, в то время как вышеупомянутого барона звали Гец.

Является ли это плагиатом? Как вообще получилось, что обнаружилась такая подозрительная схожесть между персонажем японской манги Гатсом и воспетым в европейской литературе бароном Гецем?

Конечно, это не плагиат. Гёте своими трудами сделал брутального однорукого барона участником массовой культуры. Гец становился предметом размышлений, он прочно укоренялся в ассоциативных цепочках, оказавшись в головах людей.

Разрозненные элементы из образа барона Геца кочевали из разума в разум, из идеи в идею, пока не встретились в голове молодого Кентаро Миюры, и не воплотились в Гатсе. Помимо барона, мышление Миюры объединило в себе образы средневековой Европы и породило продуманную и стильную вселенную Берсерка, элементы которой массово встречаются в серии игр Dark Souls. А уж в сознании скольких

разработчиков проросли идеи и образы Dark Souls – не пересчитать! Игр, вдохновленных этой серией, развелось немыслимо много.

Все идеи, которые мы генерируем, не берутся из вакуума. Наш мозг, подобно большой математической машине, способен лишь складывать, делить и умножать ту информацию, что попала в него извне. Прочитанные нами книги, увиденные нами картины, услышанные нами слова, пройденные нами игры и просмотренные фильмы – все это плотно утрамбовывается в нашем сознании и становится почвой для наших новых идей.

Не будете же вы спорить с тем, что у человека, лишенного медицинского образования, может появиться идея, каким же образом надо лечить, например, рак? В его голове нет знаний ни о методах лечения, ни о каких-то исторических примерах, которые могут подсказать направление работы. В свою очередь, медик, чья голова забита подобной информацией, решит вопрос с куда большим успехом.

> **Образы и знания, которые откладываются у нас в голове, формируют наши идеи. Внимательнее отнеситесь к потребляемой информации, и, если к вам не приходит вдохновения и вы не можете ничего придумать – переключайтесь на другие фильмы, игры или книги, а после этого больше времени проводите без получения новой информации в принципе.**

Перед разработкой Fearmonium я очень впечатлился увиденным на просторах YouTube фанатским клипом на песню Mercury музыканта Ghostemane. Видеорядом в нем служила весьма мрачная нарезка из старого мультфильма про клоуна Коко и Бетти Буп, а полная адреналина тяжелая электронная музыка придавала всему происходящему шарма. На досуге я читал литературу по психологии и заинтересовался становлением неврозов, а вечером решил пересмотреть свой любимый мультфильм от Disney «Пиноккио».

Таким образом, в моей голове одновременно закрутились музыка Ghostemane, классическая анимация в мрачном антураже и знания по психологии. Не удивительно, что мой мозг пришел к идее сделать Fearmonium – проект, в котором игроку выпадает роль фобии в сознании подростка. Исполнить я все это пытался в стиле Disney 1930–

1960-х годов, а сопровождать сей опус было решено мрачной электронной музыкой.

На момент появления этой идеи в моей голове витали яркие образы. Они впечатляли и вдохновляли меня. А теперь представьте, что получится, если закинуть в голову «смешные» картинки из вульгарных публичных страниц, короткие и суетливые видео из Tik-tok, глупые истории с Pikabu, однообразные матчи с онлайновых игр и сотни бессмысленных сообщений из очередного чата.

Все это будет крутиться у вас в голове, как в миксере, и, как вы думаете, какой же вид примут ваши идеи при напряженной попытке что-либо сгенерировать? В лучшем случае – никакой, идей просто не будет. В худшем – вам в голову придет мысль о том, чтобы переснять «Один дома».

Образы, крутящиеся в «миксере мышления», определяют не только ваши идеи, но и отчасти ваши желания.

Нехватка сил и мотивации при разработке игры в одиночку упоминалась как главная беда чуть чаще, чем все остальные вместе взятые проблемы разработчиков-одиночек. В формировании мотивации участвует множество процессов, но сейчас мы упомянем только один из них: вы, наверняка, не раз смотрели какое-нибудь мотивирующее видео или читали мотивирующую книгу, которая, судя по отзывам, поменяла чью-то жизнь. Страница за страницей это литературное произведение и вас наполняло энергией, вселяя стремление работать и развиваться. Вам казалось, что вы становитесь новым человеком – вдохновленным и сильным. Море будто становилось по колено.

Но проходило время. Вдохновение улетучивалось, а силы растрачивались на рутинные дела. Недавние мечты откладывались на пыльную полку, где уже заканчивается место – там пылится и без того множество старых амбиций.

Почему так происходит?

А потому, что после этого видео или книги мы снова ныряли в омут коротких и суетливых видео или разгребали болото комментариев под какой-нибудь скандальной новостью. В нашем «миксере мышления» не бесконечно много места. Мы не можем мыслить тысячей образов – у нас есть лимит. У всех он разный. Пока воспоминания о книге или видеоролике свежие – светлые идеи и мысли крутятся у вас в голове и

подталкивают к великим свершениям. Но стоит вам подкинуть в «миксер» информационного мусора, как мотивация и вдохновение заменяются чем-то в разы менее адекватным.

Информация, которую вы потребляете, определяет то, какими будут ваши идеи, появятся ли они вообще и хватит ли у вас сил на их реализацию.

Если у вас дефицит идей – попробуйте для начала просидеть десять минут в пустой комнате с выключенным компьютером и телефоном. Это даст вашей думающей части мозга отдохнуть и освободит ресурсы для той части мозга, которая занимается упорядочиванием знаний и информации. В современном ритме жизни мы отводим очень мало времени для работы механизма, который называется «архивирующим мозгом», – именно там формируются ассоциативные цепочки, способствующие появлению прекрасных идей. Эта область работает только тогда, когда вы не получаете информацию извне.

Повторить этот процесс после получения полезных или вдохновляющих знаний – вдвойне полезно.

В современном океане информационного мусора, где каждая онлайн-помойка борется за наше внимание, вынырнуть на свежий воздух и более избирательно отнестись к тому, чем мы кормим наш мозг, является задачей, решение которой требует невероятных усилий.

Но именно успешное прохождение этой непростой миссии определит гениальность нашей будущей игры.

24. Гори, но не сгорай

Иногда мы себя настраиваем ударно поработать грядущим вечером, но уже ближе к ночи на самом исходе дня обнаруживаем, что, вместо того чтобы созидать, мы провели очередной вечер за просмотром не самых полезных видеоматериалов на YouTube или за бессмысленной попыткой пролистать ленту новостей до самого конца.

Если вашим аргументом в пользу того, чтобы не заморачиваться и свободно отвлекаться на незначительную ерунду станет то, что листание новостных лент – это необходимый нам отдых, то спешу огорчить: вы понятия не имеете, что такое «отдых».

Безусловно, мы не без оснований опасаемся за свое психическое здоровье, когда много работаем и много учимся. Некоторые напуганы «эмоциональным выгоранием» настолько сильно, что почти ничего не делают, а в сторону тех, кто просиживает долгие часы за любимым делом, они бросают снисходительный взгляд и бубнят про «неизбежное перегорание». Разумеется, стоит быть аккуратным, но и излишняя жалость к себе до великих свершений тоже не доведет.

Нужно понимать, что листание ленты новостей и общение в чатах для вашего мозга являются такой же работой, как и решение сложных логических задач в ходе разработки видеоигры. Просмотр фильмов, чтение книг, компьютерная игра – все это тоже процессы, которые не позволяют включиться архивирующей части мозга. Мы продолжаем находиться в напряжении и отслеживать тысячи стимулов и сигналов в период практически любой активной деятельности.

Недостаточно просто отвлечься от разработки на вкладку в браузере со смешными картинками. Это никоим образом не спасет вас от выгорания. Но так делают почему-то все, кто действительно боится сгореть, – они «отдыхают», продолжая содержать свой мозг в тонусе и напряжении.

Выгорание по своей симптоматике очень схоже с депрессией, и многие мои коллеги частенько путают два столь малоприятных состояния. И выгоранию, и депрессии сопутствуют удушающая апатия, нехватка сил, нежелание работать, потеря веры в себя, негативные мысли, утрата мотивации к созиданию и неспособность

получать удовольствие от вещей, которые вам нравились. Даже подъем с постели превращается в настоящее испытание.

Выгорание – это не просто усталость. Если вы устали – вы все еще можете получить удовольствие от игры в видеоигры, от просмотра хорошего фильма или от времяпрепровождения с семьей. Если же вы выгорели, то любимые дела перестают приносить радость, а все желания испаряются, оставляя за собой лишь пустоту и апатию.

Исчезновение сил для работы и утрата желания созидать являются основными причинами, почему многие проекты так и не доживают до выпуска, а разработчики начинают жаловаться друг другу в потере мотивации. Скорее всего, часть этих ребят просто «сгорели».

Я не всегда был аккуратен и, разумеется, перегорел. И тогда по ошибке принял этот недуг за развивающуюся депрессию. Спрятаться от нее я пытался в работе, зарываясь в многообразие своих дел еще глубже, но тем самым я лишь усугублял свое состояние.

Мне не хотелось ни работать, ни отдыхать. Два столь противоречивых стремление сигнализировали о том, что я перегорел.

Но самое страшное и очевидное, что игнорирует перегоревший человек, – это свою тревогу. Простая усталость обращается в испепеляющее пламя выгорания только на фоне стресса. Ни один невроз не развивается без перманентной тревоги.

Сколько бы вы ни работали – вы не сгорите, если не находитесь в деструктивном стрессе и лишены тревог. Причинами тревожности могут стать не только внешние факторы вроде злых начальников и невзгод в семье, но и внутренние: высокие требования к себе, страх оказаться недостаточно компетентным, опасение стать неинтересным другим людям, боязнь быть высмеянным за свое творчество и леденящий душу ужас от возможности провалиться – все это оперативно обращает нашу усталость в выгорание.

Стресс, безусловно, не всегда бывает разрушительным – стоит отличать тип волнения, который вас стимулирует развиваться и работать, от другого типа, который портит вам настроение, отвлекает и деморализует. Это нормально – волноваться о том, как, например, встретят ваш проект игроки и не обнаружится ли кучи багов в день выхода игры. Такой стресс даже стимулирует быть внимательнее и работать лучше. Но если ваша жизнь обременена обстоятельствами,

которые вместо бодрящих всплесков адреналина только провоцируют нервозность, то тогда шансы выгореть утраиваются.

На момент, когда я выгорел, я разрабатывал Fearmonium около шести месяцев. Предыдущие проекты наградили меня слепой уверенностью в том, что у меня достаточно много опыта и умений, чтобы сделать новую игру в гораздо более сжатые сроки, чем я был на самом деле способен. В итоге через шесть месяцев у меня был готов настолько незначительный кусочек Fearmonium, что я стал себя корить за такую непродуктивность и на почве презрения к собственным умениям подгонял себя, чтобы работать все быстрее и быстрее.

Я сам создал себе стрессовую ситуацию невменяемыми требованиями, при формировании которых я не учел очевидных вещей: Fearmonium представлял собой куда более технически сложный проект, нежели все то, за что я брался раньше. Я пытался работать над детализированной покадровой анимацией с такой же скоростью, с которой я работал над примитивной костной анимацией в Reflection of Mine.

На страницах этой книги я уже давал советы о том, как важно определить свои возможности и не выполнять меньше, чем мы способны выполнить. Сейчас я снова акцентирую внимание на пределах наших возможностей и порекомендую не пытаться выполнить теперь уже больше, чем позволяют наши способности.

В формулировании предела своих возможностей стоит внимательнее концентрироваться на условиях, в которых вам некогда удалось превзойти самого себя, и стараться учесть все факторы, вследствие которых ваши трудовые будни раньше приносили больше результата, чем сейчас.

Учитывайте даже погоду! Я, например, не сразу осознал, насколько же мне плохо работается в жару, когда все, что я могу делать, это сидеть напротив вентилятора и просто таять. Иной раз я плохо работал на фоне стрессовой ситуации в другой сфере моей жизни, которая занимала меня тогда с головой и не давала погрузиться в разработку.

При попадании в новую среду стоит пересчитывать свои лимиты. Если вам, например, нужно два часа, чтобы нарисовать что-нибудь красивое в Photoshop, с которым вы работаете уже пять лет, то не рассчитывайте на то, что при работе в только что установленном на

ваш компьютер blender вам потребуется столько же времени для создания чего-то прекрасного.

Разрабатывая игру в одиночку, очень легко изменить самооценку в худшую сторону: оказавшись один на один и с кодом, и со сценарием, и с графикой, вы, вполне вероятно, где-то продемонстрируете не самый выдающийся результат. Первый показ вашей работы может спровоцировать шквал грубой критики от комментаторов в Интернете. Мнение «я ни на что не способен» лишь прочнее укоренится в вашем сознании, порождая еще больше тревоги и стресса.

Стоит иметь в виду, что оценка вашего творчества – это не оценка вас самих и даже не оценка ваших умений. Если я нарисовал плохую картинку – это не значит, что я не могу нарисовать хорошую. Это лишь значит, что конкретно эта картинка – плохая. Мы учимся. Даже выпустив одну, две, три игры, мы все продолжаем учиться, и, если мое творчество не находит отклика в сердцах достаточного количества людей, это отнюдь не значит, что я ничтожество. Я буду учиться, и все получится в другой раз. Четвертый, пятый, шестой – какая разница? Мне нравится сам процесс создания игр и процесс обучения этому ремеслу.

Решение любой проблемы – это всегда, абсолютно всегда лишь вопрос времени. Если у вас не получилось сделать красивую анимацию персонажа, это не означает, что вы бездарность и должны корить себя. Это означает, что вам нужно сделать анимацию еще раз. И еще раз. Момент, когда у вас наконец-то все получится, – лишь вопрос времени.

> **В процессе созидания вы станете автором бесчисленного множества никем не замеченных работ. Это естественно для творческой деятельности. Не судите об успехе других авторов только по играм, о которых вы слышали. Поверьте, каждый из них сделал или начал делать кучу проектов, которые так и не увидели свет.**

И если такие свойства разработки игры в одиночку, как необходимость брать на себя неимоверно много задач и возможность уронить свою самооценку ниже плинтуса, способны провоцировать выгорание, то, как ни странно, неизбежное распыление между делами

вполне может спасти нас от незавидной участи разработчиков, сгоревших в процессе созидания.

Один из способов вновь ярко вспыхнуть заключается в смене деятельности. Когда вы в командном проекте занимаетесь только, например, 3D-моделированием, взять на себя совсем иной спектр задач не всегда является возможным. Вполне вероятно, команде не требуются ваши другие, менее развитые навыки, и заниматься, например, маркетингом, вам никто не даст, потому что «вон, у нас там человек на полной ставке сидит, а ты сиди и делай модели». Сгореть в таких условиях – проще простого.

Если же вы один, то вы в любой момент можете сместить акцент своей деятельности. Я, например, так и поступил: когда мне уже второй месяц не хотелось вставать с постели и заниматься разработкой, я проанализировал то, чему я уделял внимание в последнее время и понял, что весь мой труд сводился к созданию анимаций. В Fearmonium используется классическая покадровая анимация, и неудивительно, что она отнимала у меня больше всего ресурсов. Именно она меня и сожгла.

Одним из способов выбраться из бесконечной бездны уныния стало мое переключение на проработку сценария. Я перестал требовать от себя выдачу пятнадцати кадров анимации в день и принялся проверять свои лимиты на написании монологов. Оставив в стороне осточертевшие программы для рисования, я жадно листал книги по психологии и писал фразу за фразой, терпеливо их переписывая и дополняя. Если вы зададитесь вопросом, почему в Fearmonium персонажи так много болтают, – то вот ответ. Без такого количества болтовни я бы так и оставался сгоревшим разработчиком.

У нас, одиночек, есть потрясающая возможность бросить любую деятельность, когда к горлу подберется удушающее чувство: как же мне это надоело. Если вам надоело рисовать – садитесь за код; надоело писать код – садитесь за создание диалогов; надоели диалоги – протестируйте то, что вы успели сделать или отвлекитесь на маркетинг.

Но помните: мы – не многозадачны. Не нужно делать все, что я перечислил выше, одновременно. Выполняя несколько дел якобы «в один момент», мы на самом деле просто очень быстро их чередуем между собой. На каждое «переключение» мы тратим силы и время, а

значит, уменьшаем шансы доделать свою игру и увеличиваем шансы свихнуться. Когда я говорю о смене деятельности, то подразумеваю, что вы должны отвлечься от программирования на, например, рисование, не на пятнадцать минут, а на несколько дней.

Не берите на себя слишком много. Не все аспекты игры должны быть совершенными.

Смена деятельности на ту, от которой вы получаете удовольствие, – неплохая профилактика выгорания. И те, кто с этим не согласен, могут снова задуматься о том, как обычно выглядят распространенные формы якобы «отдыха». Когда вы играете в видеоигры, расслаблен ли ваш мозг? Лично я редко бываю настолько же сконцентрирован, как при битве с очередным боссом, когда и у него, и у меня, остается здоровья на пару ударов; я редко бываю настолько же внимателен, как когда я ищу нужную мне деталь от LEGO, собирая очередную свою задумку; я очень усидчиво слушаю своих друзей, когда мы собираемся вместе и обмениваемся остротами. Мой мозг работает на всю катушку. Вы можете устать от просмотра фильмов, видеоигр или посиделок с друзьями так же сильно, как вы устаете от любого из аспектов разработки собственной видеоигры.

В случае с просмотром кино и играми вас настигнет усталость чуть позже лишь потому, что получение удовольствия от деятельности позволяет нам дольше оставаться на ней сконцентрированным: удовольствие – это отличная мотивация не отвлекаться. Занятие, которое радости нам не приносит, вынуждает мозг предлагать нам иные варианты времяпрепровождения.

Можно представить себе запас ваших сил как постепенно разряжающуюся батарейку. В таком случае получение удовольствия способно на некоторое время замедлить расход мощности, а чтобы совместить приятное с полезным – вполне можно и даже нужно научиться получать удовольствие именно от разработки видеоигр.

Здесь мы находимся в более выгодной ситуации, чем большинство людей, – мы с вами занимаемся действительно чем-то крутым и веселым, и настроить себя на получение радости в ходе процесса разработки видеоигры куда проще, чем если бы мы работали с чем-то объективно скучным. Концентрация на том, что мы создаем целый

мир, который будет радовать и вдохновлять других игроков, – лишь один из сотни способов получать удовольствие от разработки видеоигр.

Никогда не забывайте о том, почему вы сели делать игры и что вас на это вдохновило. А еще лучше – запишите это. В ходе работы над Fearmonium я несколько раз пересматривал мультфильм «Пиноккио», чтобы чувствовать стилистическую причастность к этому произведению и получать неимоверную радость от этого факта. В момент упадка сил попробуйте вернуться к тем идеям и образам, которыми вы горели и вдохновлялись, когда только начинали делать свою игру.

Концентрируйтесь в разработке на аспектах, работа над которыми делает вас счастливым. Если вы получаете безмерное удовольствие от написания диалогов – пусть диалоги станут основной составляющей вашей игры. Если вам нравится работа с кодом – вы можете добавить великое множество механик в вашу игру, а если вашей сильной стороной является изобразительное искусство, то смещайте упор на графику.

Если поставить перед собой цель, сформулированную так: «мне нужно выпустить игру», то с большой долей вероятности вы сможете в период усталости найти причины отказаться от нее, заявив себе: «Ой, да не так уж и нужно было мне выпускать игру». Если же вашей целью станет «я хочу получать удовольствие от разработки видеоигры», то выдумать себе причину «не получать удовольствия» станет потруднее.

Но, безусловно, и наша деятельность полна рутины, которая, казалось бы, не способна вызывать ничего, кроме раздражения.

Например, когда я разрабатывал Catmaze, я не очень хорошо продумал, каким же образом я буду добавлять в игру новые языки: персонажи у меня болтают достаточно много, но текст их фраз я писал прямо в «коде», а не в каком-то отдельном документе, который легко было редактировать. Для добавления новой фразы мне требовалось скопировать ее из таблицы с переводами, найти нужную строчку в коде и вставить туда текст, поменяв значение переменной, отвечающей за языки. Так делать нельзя. Но я был новичком, когда придумывал эту кривую систему и если бы я стеснялся ошибаться, то мои персонажи вообще бы не разговаривали.

У меня в игре двадцать тысяч фраз на разных языках. В определенный момент мне добавили пять новых языков, и на меня свалилась обязанность сто тысяч раз повторить одно и то же действие: скопировать строчку текста и вставить ее в нужное место. Звучит как сущий кошмар. Создать какой-нибудь алгоритм для выполнения этой чудовищно рутинной работы я не мог, потому что в моем отвратительном коде все было не так просто, и некоторые фразы приходилось копировать с «особенностями» и дописывать некоторые вещи руками.

Я был не очень хорошим программистом и обрек себя на утомительную рутину. Смог ли я получить от нее удовольствие? Да, смог, и сейчас расскажу, как.

25. Отдых важнее работы

Получение удовольствия от работы – один из способов расходовать силы куда медленнее коллег, которые не умеют «тащиться» от своей деятельности. Есть много трюков и способов, но в первую очередь я упомяну тот, который лично мне помог разобраться с проблемой длиною в сто тысяч строк.

Представьте, что какое-то время вы, проживая в гордом одиночестве, решили вести образ жизни холостяка. Вы не занимались домом, постоянно ели пиццу, раскидывали кругом грязную одежду и пустые коробки из-под сока. Вы уже какое-то время не видели в этой горе мусора своего кота, но вас это не беспокоит, потому что его миски постоянно пустеют, а значит, он или где-то прячется, или у вас завелось какое-то другое животное.

Но вот вам звонит ваша дальняя родственница, от которой вы отнюдь не в восторге, и напоминает вам о том, что вечером она будет у вас. Вы обещали обеспечить ей ночлег, но благополучно об этом забыли. Впускать живого человека в обитель лени и хаоса – бесчеловечно! Вам придется навести здесь порядок.

С каким настроением вы будете это делать? Вам приходится суетиться из-за забытого вами обещания принять гостью и тратить свой выходной на уборку, которую вы совсем не планировали. Разумеется, вы будете недовольны.

А теперь отмотаем время назад и представим, что вместо неприятной родственницы вам поступил звонок от человека, на романтические отношения с которым вы давно рассчитывали. Абсолютно внезапно вы получаете ответ на ваши бесчисленные знаки внимания, и свидание состоится сегодня вечером. У вас дома.

С каким настроением вы будете наводить чистоту теперь? Те же самые действия, которые в предыдущем примере могли вызвать только негодование и уныние, теперь уже будут сопровождаться легкими размашистыми движениями и напеванием какой-нибудь хорошей песни себе под нос.

Вы знаете, что изменилось? Контекст. Вы все так же убираетесь в квартире, просто теперь у вас иное понимание данной ситуации. Наше

отношение к той или иной деятельности полностью зависит от контекста.

Управлять им можно даже мысленно. Когда я думал, что я вставляю сто тысяч строк текста ради денег, которые я получу из других регионов, или ради издателя, который меня попросил это сделать, – на меня нападало уныние. Деньги и желание угодить другим людям никогда особо меня не стимулировали чем-либо заниматься.

Но затем я представил, как какие-нибудь немецкие мальчишки сидят во дворе со своими приставками Nintendo и обсуждают сюжет Catmaze. Они используют немецкий язык, и если бы я его не вставил в игру, то они бы не смогли поиграть в нее в принципе. Тогда у меня появились силы. Я сам придумал себе контекст, в котором я копирую и вставляю строчки текста ради того, чтобы мои игры смогли порадовать большее количество людей по всему миру. Чтобы над моими шутками посмеялись японцы, чтобы характер моих персонажей оценили итальянцы, чтобы ребята из Франции спорили, какой фамильяр у Алесты самый крутой. Я стал получать удовольствие от скучного, но уже отнюдь не изнурительного дела.

> **Думайте о контексте, в котором вы работаете. Мысленно регулируйте ваше представление об обстоятельствах своей деятельности таким образом, чтобы она начинала вас радовать.**

Секрет экономии сил во время умственного труда кроется в получении от него удовольствия. Чем больше вас радует процесс разработки, тем боле шансов, что вы предпочтете делать игры вместо того, чтобы в них играть.

Но тем не менее я не сторонник современного культа продуктивности, и, если вы думаете, что, даже получая от работы радость, вам удастся вкалывать по сорок восемь часов в сутки и не устать – вы ошибаетесь. Силы все равно тратятся. Удовольствие лишь замедляет этот процесс. Необходимость отдыхать «по-человечески» ничем не заменить.

Первый важнейший аспект для восстановления сил – это, разумеется, здоровый сон. Вы наверняка натыкались на информацию, что человек должен спать семь-восемь часов в сутки, но сами же вряд

ли следуете этим заветам. На самом деле необходимое для хорошего самочувствия время сна индивидуально и зависит от количества психической энергии, что вырабатывается нашим мозгом. Семь-восемь часов – это «средняя температура по больнице», в которой значение температуры и лихорадочных больных, и трупов в морге в среднем составляет 36,6 градуса.

Прислушайтесь к своему организму и выявите самостоятельно, сколько сна вам требуется для поддержания энергичного образа жизни. Многим из нас кажется, что люди, которые спят по четыре часа, располагают более внушительным запасом времени, который они используют для свершения великих дел. «Цезарь же спал по четыре часа и именно поэтому сколького добился, а я вот не высыпаюсь и потому ничего не успеваю». С такими мыслями люди решают или опустить руки, или спать поменьше.

Во-первых, освободившееся благодаря недосыпу время не имеет ценности: вы не набрались сил. Без сил реакции и решения будут заторможенными. Во-вторых, лично я сплю по девять-десять часов и успеваю делать очень-очень много – куда больше, чем я успевал в студенческие годы, когда спал по пять-шесть часов. Мы все сосредоточены на том, как бы пораньше проснуться. Но причина, по которой так тяжело начать новый день, кроется не в том, как мы просыпаемся, а в том, как мы засыпали. Заснуть вовремя куда важнее и сложнее, чем вовремя проснуться.

Заканчивать активную умственную работу нужно хотя бы за час до того, как вы собрались спать. Вам нужно выйти из режима генерации идей, позаниматься рутинными делами не за экраном монитора, а лучше – отключиться от Сети. Так вы лишите свой рефлекторный мозг стимулов, которые способны вынудить вас начать сосредоточенно читать письма и новости и возвращаться в состояние возбуждения, в котором заснуть очень сложно.

Я частенько слышу от людей, что они не высыпаются, а потом узнаю, что они ложатся спать, не выпуская из рук смартфона и не снимая наушников. Они вынуждают мозг всю ночь искать стимулы и изменения в окружающих звуках и изображениях; он продолжать усердно работать вместо того, чтобы разгребать завалы полученной за день информации и составлять ассоциативные цепочки.

Я не удивлен, что такие люди не могут выспаться. Совет не засыпать, будучи подключенным к Сети через свой смартфон, – это очевидный и прямолинейный совет.

А вот чему я действительно удивлен, так это тому, как люди превращают в работу то, что, казалось бы, должно расслаблять. Простая игра в видеоигры сводится у кого-то к бездумному набиванию достижений, и важным становится не процесс игры, а результат. Подобным образом у кого-то даже работает чтение книг – люди ставят перед собой цель прочесть определенное количество книг за год, в то время как литература создана для того, чтобы мы получали от нее удовольствие, а не гнались к каким-то измеримым целям.

Я понимаю, как достижение результатов может приносить удовольствие, но разрушительность соревновательного подхода к отдыху заключается в том, что вы продолжаете устанавливать перед собой какие-то требования, а пока у нас есть требования к себе – у нас есть стресс.

Стресс – главный противник отдыха. Вы не наберетесь сил, оставаясь в тревожном состоянии.

Многие, кто сталкивался с такими изматывающими трудностями, как, например, неудачный запуск игры, вполне оправданно пытались спрятаться от выгорания в коротеньком отпуске. Они собирали вещи, ехали на море, смотрели на волны и… думали о том, что их игра провалилась.

Мы слишком часто берем стресс с собой в отпуск. Если во время отдыха вы продолжаете думать о провалах, грядущих испытаниях или прочих невзгодах – это не отдых. Вы продолжаете гореть.

Бесспорно, прогулка по парку без наушников и смартфона в кармане – это прекрасная форма отдыха, позволяющая вашему мозгу разобрать все завалы из информации, которую вы туда небрежно накидали. Но если сию секунду последовать этому совету, накинуть куртку и побежать в парк, то, скорее всего, все ваши тревоги и мысли пойдут вслед за вами. Вместо того чтобы предоставить мозгу достаточно ресурсов для архивирующей работы, вы отдадите все свои силы думающему мозгу, вынуждая его гонять мысли в духе «надо было ответить иначе на тот комментарий!» или «а что, если я поменяю в своей игре главного персонажа?»

Все наши мыслительные ресурсы отправятся на жевание этой жвачки, и отдохнуть снова не получится. Наши проблемы продолжают находиться в «миксере мышления», и мы в целом разобрались, каким образом туда попадают мысли и образы. А вот о том, как их оттуда вытаскивать, написано множество отличных и очень толстых книг, но иной раз и сотни страниц недостаточно для того, чтобы сформировать необходимое для «очищения миксера» убеждение. Передо мной же стоит цель уложиться в пару абзацев.

«Не думать о белой обезьяне» – сложная задача, и для ее выполнения я порекомендую вам сейчас отвлечься. Посмотреть вокруг себя, выглянуть в окно, перевести взгляд на свежую листву или на бескрайнее небо, а все внимание направить на то, что происходит прямо сейчас, в этот самый момент.

Каждый миг – уникален. Не рассматривайте его как нечто, что влияет на будущее или сформировано прошлым: для очищения мыслей нужно находиться здесь и сейчас.

Мысли о прошлом могут породить грусть, мысли о будущем – тревогу. Чтобы отдохнуть от всего этого, нужно осознать себя именно в тот момент, в котором мы находимся сию минуту. Это схоже с медитацией, но вполне достижимо и без познаний в загадочных для нас техниках.

Результаты настоящего отдыха без подключенности к сети, без жевания мыслительной жвачки и в абстрагировании от стресса воистину впечатляющи.

Я осознал все это в период, когда я участвовал в том самом моем первом джеме. Времени у меня было меньше, чем у многих, потому что на носу висели кандидатские экзамены и, как назло, навалилось несколько ночных смен на работе.

Меня охватывало отчаяние, когда я бился об очередную проблему в игре, словно рыба об лед, и не мог придумать решения целую бессонную ночь. Проснувшись после неудачных попыток реализовать желаемое, я планировал один из немногих свободных дней целиком посвятить разработке игры, но суровая действительность ополчилась против меня: сразу с утра я обнаружил, что в квартире отключили свет.

Как ни странно, вряд ли можно назвать более значимое событие в моей карьере, нежели тот инцидент со светом. Важность вещей, которые я тогда осознал, неоспорима.

Во-первых, у меня никогда не было и до сих пор нет смартфона – так что, лишившись доступа к стационарному компьютеру, я остался без подключенности к сети. Во-вторых, я был вынужден погрузиться в рутину домашних дел в полном отсутствии каких-либо стимулов к другой активности: я не мог получить оповещений в браузере, у меня не работал телевизор и даже послушать музыку мне было не на чем. Обыденные дела вынудили меня сосредоточиться на настоящем и на какое-то время забыть о грядущих экзаменах, об изматывающих проблемах на работе и даже о грядущей игре.

Я оказался в ситуации, в которой я ничего не мог сделать для своего будущего.

Чувство тревоги, обиды и откровенной «ломки» по работе с компьютером меня разъедало, наверное, около часа. Но произошедшее чуть позже обогатило меня практическими знаниями о том, что такое дофаминовая зависимость и из какого же хлама, волнений и сетевого мусора состоял мой «миксер мышления». Когда я уже просто лежал на полу и ждал, пока включат свет, на меня нахлынул поток идей, вдохновения и способности концентрироваться на настоящем.

Восприятие мира, наступившее после информационной разгрузки, оказалось таким, каким оно было для меня двадцать лет назад, когда я был ребенком. Все стало интересным. Я мог концентрироваться на чем угодно сколь угодно долго. Проблемы с разработкой игры я решил прямо у себя в уме. Знания, необходимые мне для успешной сдачи экзамена, наконец-то упорядочились, и вспомнить что-либо из необходимых мне материалов стало куда проще.

Когда свет все-таки включили, я потратил две минуты на решение тех проблем, что прошлой ночью отняли у меня несколько часов моей жизни. Набравшись сил и идей, я чувствовал себя бодрым и активным подростком.

Вот какие способности и эмоции дает настоящий отдых. Не смена деятельности, не досуг, а именно отдых. Простое лежание на полу без света можно заменить занятием спортом, пребыванием в душе, прогулками, да просто любым медитативным времяпрепровождением, разгружающим ваш думающий мозг и позволяющим концентрироваться на настоящем.

Множество идей для моих игр пришло ко мне на пробежках, некоторые – посетили меня пока я бездумно стоял под струями воды в

душе. Мой мозг не является каким-то уникальным в этом плане: мышление любого из нас будет работать точно так же.

> **Отдых и стресс являются несовместимыми вещами. Учитесь жить настоящим моментом; думать о том, что окружает вас прямо сейчас; концентрироваться на вот этой конкретной секунде. Спрятаться от стресса можно только в текущем мгновении.**

Мы же все знаем, что закон всемирного тяготения Ньютон открыл не во время кропотливого и сосредоточенного анализа научных данных, а во время отдыха под деревом. Сосредоточенность на настоящем позволила ему обратить внимание на падающее яблоко, а мозг уже успел упорядочить достаточное количество знаний и фактов, чтобы сложить их все вместе и прийти к выводу о существовании всемирного тяготения.

А таблица Менделеева, увиденная во сне? А открытие Архимеда, совершенное им в ванне? Наша массовая культура пропитана примерами, когда отдых освобождает великие умы от суетливого информационного хаоса и предоставляет им возможность совершить грандиозное умозаключение.

Так по какой же причине мы не позволяем себе впасть в такое же состояние просвещенности? Почему мы считаем необходимым быть постоянно подключенными к Сети и находиться под непрекращающейся бомбардировкой информационного шума? Почему так трудно расслабиться и сосредоточиться на настоящем?

Наша зависимость от сиюминутных удовольствий, поджидающих нас в Интернете, загоняет наш потенциал в невероятно узкие рамки. Листание ленты и активная переписка в многочисленных чатах невероятно далеки от состояния покоя, в котором наши силы на самом деле восстанавливаются. Когда мы отвлечены на потребление информации – наш мозг не отдыхает, а продолжает работать на всю катушку. Не надо путать любование солнышком и настоящими бабочками с просмотром коротких и суетливых видеороликов в Tik-tok. Но почему же нас тянет именно в Сеть?

Отвлекает нас от работы и вынуждает в очередной раз проверить почту та часть мозга, которая называется «рептильной», или

«рефлекторной». Рефлекторный мозг – это самая дикая, жадная и примитивная часть мозга, которая некогда обеспечивала нам выживание, принимая быстрые решения «убежать от того гепарда» или «спрятаться от медведя на дереве». Сейчас же рефлекторный мозг вынуждает нас постоянно утолять свое любопытство в бесконечных новостных лентах и проверках почтовых адресов, потому что, во-первых, там может быть угроза, которую нужно исключить, а во-вторых, там может быть и что-нибудь приятное, что вызовет кратковременный всплеск гормона эйфории – дофамина. Мозг запомнит эту реакцию и будет стимулировать нас ее повторить.

Рекомендую ли я совсем отказаться от смартфонов и подключенности к Сети? Это было бы смелой рекомендацией, и для того чтобы убедить читателя в ее правильности, мне потребуется отдельная книга. На страницах же этого издания я посоветую лишь пересмотреть свой взгляд на отдых: чаще оставаться без подключения к Сети и чаще полностью фокусироваться на настоящем.

Также я рекомендую задуматься о том, что потреблять информацию в Интернете и «залипать» в социальных сетях не является вашим полностью осознанным выбором: сотни исследований, тестов и экспериментов позволили самым популярным сервисам прийти именно к той форме подачи материала, которая не дает пользователю оторваться от экрана телефона. Бесконечная новостная лента; краткость постов; емкое расположение картинок; хитрые алгоритмы, подсовывающие вам новости и знающие о ваших интересах больше, чем кто-либо, – все это выступает в роли «гипноза», превращающего пользователя в зомби. Только пищей выступают не мозги, а информация. Чаще всего, кстати, совсем бесполезная.

Абсолютно никто «по ту сторону экрана» не заинтересован в том, чтобы вас развлекать: им нужны ваши просмотры, лайки и активность, потому что все это увеличивает ценность сервиса, где вы залипли, и повышает стоимость рекламы на нем.

Я нередко слышал истории про то, как переезд в страны Европы вынуждал моих коллег возвращаться к использованию низкоскоростного подключения к Интернету. Несмотря на то, что ухудшение условий работы в Сети кажется объективно неприятным событием, через некоторое время ребята вырабатывали более избирательный подход к потреблению информации, избавляясь от

привычки бездумно листать ленту новостей и смотреть ролики на YouTube. Еще бы! Ведь лента еле грузилась, а смотреть каждый якобы интересный видеоролик не позволял ограниченный трафик. Время в Сети перестало восприниматься как отдых, и чтобы расслабиться парни отвлекались на что-то, что с бóльшей вероятностью приводило их ум в порядок.

Но, увы, совсем Интернет нам не покинуть: нам необходимо заявлять в нем о себе. Тем более многих разработчиков преследует один и тот же страх: страх быть незамеченным и не продать достаточное количество копий игры. Работать над играми в страхе и стрессе – не лучшее решение.

Стресс – это неадекватное восприятие жизненных обстоятельств, но наличие программы действий по продвижению и распространению своей игры должно успокоить нашего внутреннего монстра.

26. Юридические тонкости

Забота о маркетинге в тот момент, когда ваша игра едва начата, может показаться не очень разумной, но знания о том, как продвигать ваш продукт в будущем, вполне могут обрисовать грядущую программу действий и избавить вас от тревожной мысли: «мою игру никто не увидит».

Ее и правда никто не увидит, если вы не приложите для этого определенных усилий. Мы же с вами разработчики-одиночки, так что обязанности отдела маркетинга также лежат на нас.

Самой развитой на данный момент площадкой для распространения независимых игр является интернет-платформа Steam, с которой, я надеюсь, вы знакомы хотя бы как игрок. Другие площадки, такие как EGS или GOG, представляют собой более закрытые платформы, выход на которые не автоматизирован, а осуществляется в ходе налаживания личных контактов. Несмотря на завышенную сложность выхода на EGS или GOG, статистика показывает, что более 90 % дохода большинства разработчиков, выложивших игру во все три магазина, приходится все-таки на Steam, оттого говорить мы будем в первую очередь про эту платформу.

В 2021 году количество активных пользователей Steam превышало 120 миллионов, а прямо сейчас, одномоментно, платформой пользуются 20–25 миллионов игроков по всему миру. При правильном использовании инструментов, которые предоставляет эта площадка, у вас появляется выход на немыслимо огромную аудиторию.

Все начинается с регистрации на partner.steamgames.com. Вы можете использовать и свою учетную запись «игрока», но я настоятельно рекомендую завести себе отдельный аккаунт: когда вы выпустите свою игру, некоторые игроки захотят добавиться вам в «друзья» или для обсуждения каких-то проблем в игре, или для простого общения. Не думаю, что им стоит видеть, как часто вы запускаете игры из своей библиотеки, в то время как в вашей собственной игре не все проблемы решены.

Регистрация партнерского аккаунта потребует немало терпения, а бóльшее количество соков из вас выпьет чудовищная форма W8BEN, в

которой нужно всеми силами доказать, что вы не являетесь резидентом США и вообще не претендуете на резидентство там. По окончании заполнения вы должны увидеть оповещение, что ваш налог в США составит 0 %. Если же вы увидели сумму больше, то это значит, что где-то вы ошиблись и лучше заполнить форму заново. По запросу «W8BEN инструкция» в Google вы сможете найти объяснения для некоторых самых малопонятных пунктов.

Второй проблемой станет вопрос о форме регистрации уже в РФ: многие разработчики весьма рано начинают думать о том, как регистрироваться в нашей местной налоговой. До получения первой прибыли от игры нет абсолютно никакого смысла менять статус физического лица на юридическое лицо и забивать голову лишней бумажной волокитой. Регистрируясь в Steam, вы вполне можете в финансовой информации указать банковские реквизиты вашего долларового счета, который вы открыли как физическое лицо.

Статус индивидуального предпринимателя (ИП) или самозанятого потребуется тогда, когда вы начнете получать прибыль. В случае если вы останетесь физическим лицом и получите деньги со Steam, вам, по законодательству РФ, придется самостоятельно платить 13 % подоходного налога, а вот в случае получения средств на счет ИП и постановке его на упрощенную систему налогообложения (УСН) подоходный налог снизится до 6 %.

Современные банковские системы позволят вам открыть ИП и поставить его на УСН, вообще не выходя из дома. С открытием ИП не стоит торопиться из-за того, что наличие ИП накладывает на вас обязанность выплачивать фиксированные страховые взносы каждый год в размере приблизительно сорока тысяч рублей. Сумма уплаченных взносов будет вычтена с суммы оплаченных налогов. Иными словами, если 6 % от вашей прибыли составили пятьдесят тысяч рублей, а вы уже заплатили фиксированные взносы, то в налоговую вам останется донести десять тысяч рублей. Это – основная причина, почему нецелесообразно открывать ИП до момента, перед которым вы начнете получать прибыль.

Про «самозанятость» рассказать сложнее всего, потому что и налоговая ставка, и условия, и даже возможность зарегистрировать себя в качестве самозанятого разнятся от региона к региону. В большинстве регионов России налоговая ставка самозанятого еще

ниже, чем у ИП, но сумма, которую позволительно заработать в этом статусе, разнится. В случае успеха вашей игры вам все равно придется становиться индивидуальным предпринимателем.

Для регистрации выбирайте банк, который может получать платежи из-за рубежа: Steam будет отправлять вам деньги Swift переводом в долларах, а в наши дни не все банки способны принять такой платеж. Постоянно обновляемые списки банков, не попавших под санкции и принимающих Swift-платежи, можно легко найти в Интернете. Но прошу, обращайте внимание на условия обслуживания валютных счетов и на комиссии за получение валютных платежей. У некоторых банков они стали абсурдно высокими.

Третья сложность при регистрации в Steam может возникнуть на этапе оплаты Steam Direct. По правилам платформы, за каждый публикуемый продукт вы обязаны внести сумму в размере ста долларов – это и называется «покупкой Steam Direct». Сто долларов являются своеобразным залогом, потому что эти деньги вернутся к вам после того, как ваш проект заработает первую тысячу долларов. Если все делать правильно, это значит, что деньги вернутся к вам уже в первом месяце.

На момент написания этой книги оплата Steam Direct была затруднена усложнением международных переводов из России. Пополнить кошелек Steam и купить там игру, находясь в РФ, всегда можно было и без Visa и MasterCard – например, с помощью сервисов Qiwi и Webmoney, но вот оплатить через эти сервисы Steam Direct на данный момент невозможно. Разработчикам приходится изобретать велосипеды и искать разные методы оплаты. Каждый из методов завязан на наличии или знакомых за границей или хотя бы банковской карты другой страны, оформленной абсолютно на любое имя.

Ваш аккаунт разработчика Steam позволяет вам добавлять зарегистрированных в Steam пользователей как своих «сотрудников», потому что в крупных компаниях доступ к партнерскому аккаунту должен быть явно не только у одного человека. Таким «администратором» можно сделать любого пользователя Steam, выслав ему соответствующее приглашение. Если добавить в свой партнерский аккаунт «администратора», чей аккаунт не привязан к российскому региону, и обеспечить его правом распоряжаться

финансовой информацией, то ваш новоиспеченный коллега вполне сможет оплатить Steam Direct.

Второй способ оплатить Steam Direct в суровых условиях нашего временного континуума заключается в работе с западным партнером, у которого уже есть партнерский договор со Steam и который уже публиковал игры на этой площадке. Он может купить Steam Direct, находясь у себя в стране, и с помощью внутреннего сервиса App Transfer передать абсолютно весь контроль над купленным приложением вам. Для того чтобы найти такого человека, можно воспользоваться советами из предыдущих глав и посетить встречу разработчиков игр, где сарафанное радио выведет вас на нужных людей.

Оба названных мною способа на момент написания этих абзацев не противоречили правилам площадки Steam и не раз использовались разработчиками видеоигр из РФ.

При подборе цены на ваш продукт стоит отталкиваться, в первую очередь, от ценников, установленных конкурентами. Вам придется поискать на площадке похожие игры, посмотреть сколько они стоят и, отталкиваясь от полученной информации, определиться со стоимостью собственного продукта. Не важно, сколько вы хотите заработать: если ваши прямые конкуренты предлагают игрокам в разы более выгодное предложение, то вы не заработаете ни копейки.

Не лишним будет знать о наличии региональных цен. Региональная цена – это не просто прямая конвертация валюты по текущему курсу. При установке региональных цен учитывается множество факторов (например, средний доход населения), и таким образом ценник $10 в США превращается в 385 рублей в России и в 37 юаней в Китае, так что средняя цена на игру составит отнюдь не $10, а чуть ли не на треть меньше. Но даже от купивших вашу игру за десять долларов вы не получите всей суммы: не стоит забывать, что комиссия Steam составляет около 30 %. В эту внушительную сумму входит множество мелких платежей вроде комиссий банкам и затрат на переводы.

А еще игроки будут делать «возвраты» вашей игры. Опция «вернуть деньги за игру» доступна тем, кто не наиграл в нее больше двух часов и купил игру менее двух недель назад. Таких ребят всегда будет около 10–15 %. Что самое забавное, некоторые из них обязаны указывать причины, по которым они вернули игру, но не все из игроков знают,

что разработчик эти причины видит в специальной закладке на партнерском сайте. Так, например, мою игру Catmaze вернули, указав причиной «я думал, что я играю в GTA», а просьбу вернуть деньги за Reflection of Mine от человека, наигравшего аж шестнадцать часов, удовлетворили из-за того, что он написал: «Я не играл – я просто уснул за компьютером».

В итоге почти половину стоимости вашей игры растаскают различные комиссии, издержки и налоги (30 % Steam + 10–15 % возвраты + 7–13 % налоги). Но не стоит печалиться: куда разумнее будет воспринимать эту ситуацию не как вопиющую несправедливость, а как плату за возможность получить доступ к многомиллионной аудитории Steam, потому что без подобных площадок лично я не заработал бы вообще ни цента. Так что лучше 50 % от какой-нибудь суммы, чем 100 % от нуля.

Разработчик оригинальной Prince of Persia – одной из самых гениальных игр из всех когда-либо созданных – отдавал со своего дохода 75 %, а перспективная французская студия In Utero разорилась из-за сделки с UbiSoft, занимавшейся издательством игр от этих талантливых ребят: их игра Evil Twin по неуклюже составленному договору должна была приносить прибыль самой студии только после того, как игра заработает больше трехсот тысяч долларов. Пока прибыль неторопливо шла к UbiSoft, штат разработчиков растекался, и студия, выкинув на рынок еще парочку слепленных на скорую руку игр, с треском развалилась.

Когда триста тысяч долларов были заработаны, в In Utero уже почти не осталось сотрудников.

> **Не спешите заморачиваться с регистрацией в налоговой службе. До того как вы начали получать первый доход, вы можете использовать реквизиты своего личного банковского счета.**

27. Издатели

Вот мы и перешли к теме издателей, затронув ее в мрачном примере сотрудничества UbiSoft и In Utero. Рано или поздно многие из разработчиков задумываются о взаимодействии с издателями, которых в наши дни развелось чуть ли не больше, чем самих разработчиков. Еще бы их было мало – глупо упускать возможность заработать денег на игре, которую ты не разрабатывал. Издатели могут иной раз забирать до 80 % прибыли и очень редко они берут меньше 30 %. Чаще всего в договорах фигурируют цифры в 30–50 % от вашего дохода.

Издатели предлагают ряд услуг, которые иной раз очень хочется переложить на чужие плечи: локализацию, маркетинг и возможность получить «финансирование». Все это можно назвать бизнес-составляющей разработки игр, и эти сферы очень плохо соотносятся с творчеством и реализацией своих идей.

Я сам издаю чужие игры на Nintendo Switch, помогая ребятам преодолеть десятки барьеров между их проектом и консольным рынком. Фокус в том, что для разработки игры для ПК вам нужен только компьютер, а для выпуска новоиспеченного шедевра в Steam – доступ в Интернет и сто долларов, которые и то к вам вернутся. Для выпуска игры на консоли вам нужны танец с бубном, шаман, хитрость и удача.

На данный момент актуальными консолями являются Playstation 4/Playstation 5, Xbox Series и Nintendo Switch. Для каждой из них нужен свой волшебный бубен, но основной проблемой всегда является получение такой вещи, как DevKit. Дело в том, что на купленной в магазине консоли вы не сможете запустить разработанное вами приложение, а вот на версии этой же консоли для разработчика – сможете. Именно такие версии и называются DevKit, и в их получении кроется самый главный подводный камень.

Во-первых, у Sony и у Nintendo для того, чтобы получить доступ к кнопке «заказать DevKit», вам придется пройти через процедуру подтверждения вашего аккаунта, для которой нужно продемонстрировать, какие игры вы вообще собрались выпускать. Если модераторам Sony эти игры понравятся, то вы сможете заплатить

за DevKit, заказать его доступ в Россию, а потом получить оповещение от таможенной службы РФ о том, что ввоз устройств, использующих шифрование данных, запрещен Таможенным союзом. Я знаю много разработчиков, чьи оплаченные DevKit'ы благополучно отправлялись обратно в Европу.

Если вам пришло в голову заказывать DevKit на чужой адрес за границей, то сначала придется попытаться сделать вид, что это ваш юридический адрес, на который зарегистрирована и ваша компания. Обойти эту систему можно разными способами, но подавляющее большинство разработчиков не захочет настолько сильно усложнять свою жизнь и лучше уж прибегнет к помощи издателей, у которых DevKit'ы уже есть.

Во-вторых, помимо самого DevKit'а вам нужно будет найти способ экспортировать проект в требуемый для определенной консоли формат. Практически любой движок без проблем соберет вашу игру в файл запуска. exe, который прекрасно запустится на любой машине под управлением Windows. Вы можете даже не заморачиваться с форматами файлов и не знать ничего про. exe, ибо любой современный движок все сделает за вас и без вашего участия.

Но. exe не запустится ни на одной консоли: каждая из них поддерживает свой формат, и второй бубен вам пригодится как раз для того, чтобы из своего движка «извлечь» игру в нужном формате. Доступ к инструментам для данной процедуры вы получаете после того, как компания, выпускающая консоль, одобрит вашу учетную запись и строго оценит игры, для выпуска которых она сейчас выдает вам необходимые инструменты.

Стоит иметь в виду, что условия использования этих инструментов у всех движков разные и, более того, постоянно меняются. Раньше самое простое решение было предоставлено для Unity, и заключалось оно в установке дополнительного модуля на движок, который добавлял кнопку «export to.nsp» (.nsp – это формат игр для Nintendo Switch), но чуть позже стало требоваться наличие платной подписки на Unity. Ее, кстати, может предоставить издатель.

Несколько лет назад подобный модуль для движка GameMaker стоил шестьсот долларов в год, но сейчас его цена, как ни странно, стремительно падает. Пальму первенства в простоте экспорта пока что держит Unreal Engine, доступ к дополнительным модулям на который

предоставляется бесплатно после прохождения всех проверок от вышеназванных компаний. Проблемой станет оптимизация игр – не так-то легко будет заставить игру, разработанную на Unreal Engine, работать без «тормозов» на Nintendo Switch. Если с самого начала разработки вы не задумывались о выходе на эту платформу, то вас ждет невероятное приключение, состоящее из бесконечного упрощения и исправления всего, что можно упростить и исправить.

Для движка Construct никаких подобных решений не существует в принципе, оттого мне как разработчику на этом движке приходится работать с группой программистов под началом издателя из Испании, чтобы выпустить свои игры на консоли: какой низкоуровневой магией они занимаются, чтобы конвертировать мои игры в нужный для консолей формат, я не имею ни малейшего представления. Обязанность портировать игры – это то, что я решил делегировать на других людей.

Мало кто вступает на дорогу разработчика игр, желая заниматься именно раскруткой своего продукта. Как раз напротив, повышенную тревожность у разработчиков вызывает именно незнание того, как свой продукт продвигать и как найти игроков. Выпуск игры без издателя и его информационной поддержки кажется чем-то вроде игры в русскую рулетку: начинающие разработчики часто разрабатывают свои проекты, не будучи способными просчитать и оценить их успех.

Исходя из этого идея получить финансирование кажется многим крайне привлекательной. Но помните две вещи: во-первых, никто никогда не даст вам денег без расчета получить обратно бóльшую сумму. Инвестиции нужно возвращать – как в форме дополнительных процентов на прибыль поверх ваших 50 %, так и в форме изначально установленной ставки – издатель будет продолжать получать прибыль с вашей игры, даже когда его затраты окупятся. Во-вторых, убеждение, что игру можно сделать, только располагая большим бюджетом, ошибочно. Наличие внушительного бюджета лишь увеличит ваши риски, повышая требования к сумме, которую должна заработать игра, чтобы окупиться. Если у вас мало опыта, то, когда вы получите пять миллионов долларов, вы просто сделаете плохую игру за пять миллионов долларов. Сделайте сначала плохую игру без затрат, а после этого уже подумайте об инвестициях.

Разработчики прячутся под крыло издателей в страхе провалиться. Когда мы находимся за спинами знающих свое дело людей, нам кажется, что почва под ногами становится надежнее и крепче. Если говорить только о маркетинге, то кажется очень соблазнительным выпустить игру под внимательным надзором таких массивных издателей, как Square Enix (серия Final Fantasy, Lara Croft), Bandai Namco (серия игр Dark Souls, Elden Ring, Little Nightmares), или независимых Devolver Digital (Hotline Miami, Loop Hero).

Но давайте попробуем вспомнить игру Hexodius. Слышали ли вы хоть раз это название? Я сомневаюсь. За девять лет существования игра заработала тридцать четыре отзыва в Steam; а если речь идет о первых пяти сотнях отзывов, то их оставляет каждый шестидесятый игрок. Злосчастную игру Hexodius купили приблизительно две тысячи человек. Это откровенно можно назвать провалом, но любопытен этот провал тем, что издатель Hexodius – это внушительных размеров компания Bandai Namco.

Можно еще поговорить об игре Octathedron, которая за четыре года собрала около восьмидесяти обзоров, спрятавшись за спинами аж самих Square Enix, неспособных в данном случае обеспечить игре достойные продажи. Копеечная игра Umiro за четыре года собрала около ста отзывов, несмотря на то что на рынок ее выводили Devolver Digital.

Я не спешу утверждать, что все издатели плохие и работать с ними не стоит. С кем работать – решать вам. Я лишь хочу развеять слепую уверенность в том, что маркетинг со стороны издательства гарантирует вам успешные продажи. Как мы видим из приведенных выше примеров, это далеко не так. Вы сами можете зайти в Steam, найти там игру, выпущенную любым известным издателем, на странице с игрой отыскать графу «издатель», нажать на указанное там название и увидеть список всех изданных им игр. Среди хитов, сделавших имя издательства узнаваемым, вы найдете десятки игр, обреченных на существование в неизвестности.

Чтобы понять, почему так происходит, нужно разобраться в самом подходе издательства к маркетингу. Мало какие компании имеют свой штат маркетологов. Чаще всего они обращаются к уже сформированным маркетинговым студиям, которые вы легко найдете в поисковике по запросу «indie game marketing agency». Редко

афишируется, раскруткой каких игр занималось то или иное агентство, потому что издатель заинтересован в популяризации своего бренда и у него есть тысячи причин, чтобы оставить вышеупомянутые кампании в своей тени.

В большинстве случаев маркетинговые агентства работают за единовременную выплату, в то время как издатель берет свой процент с вашего продукта на протяжении бесконечности.

Работа издателя выглядит в большинстве случаев примерно так: предположим, что он заключил договор с двумя разработчиками на две разные игры, ценник на которые будет примерно одинаковым. Сначала вкладывается одинаковая сумма в раскрутку каждой из этих игр. Предположим, по пять тысяч евро на каждый проект. Маркетинговое агентство, получив деньги, занимается рассылкой писем, презентацией игры, созданием пресс-кита (набор рекламных материалов для рассылки журналистам). Когда их работа выполнена, издатель может оценить результаты используя или свою аналитику, или элементарную статистику, которую предоставляет Steam: еще до выхода игры можно выяснить, сколько у вас заинтересованных в покупке игроков.

Если при одинаковых расходах на маркетинг у одной игры аудитория растет гораздо быстрее, чем у второй игры, то издатель, занимающийся бизнесом, а не творчеством, перестает выделять деньги на маркетинг второй игры: ему это не выгодно. Лучше еще больше денег вливать в раскрутку того проекта, которым аудитория заинтересовалась сильнее.

Ни в одном договоре с издателем не будет указана фиксированная сумма, которую он должен потратить на маркетинг вашего проекта. Если он захочет, то потратит сотни тысяч долларов, а если нет, то вложит в раскрутку пару баксов, а потом ваша игра будет лежать в дальнем ящике, забытая и ненужная. Бизнесмен будет просто ждать, пока вы ее доделаете, провалитесь, а ему принесете ну хоть какую-нибудь копеечку, которая, может, даже не окупит затрат на локализацию (перевод игры на другие языки).

Издатель может обеспечить успешный выход вашей игры. А может и не обеспечить. Ровно как и вы: возможно, вы сами раскрутите проект достаточно хорошо, а возможно – и нет. Наличие издателя не сокращает вероятность провала, но

полный контроль над выпуском приложения возможен только тогда, когда вы занимаетесь выпуском игры самостоятельно.

28. Учим все языки на свете

Мы разрабатываем игру в одиночку и перевести ее мы тоже должны самостоятельно, оттого рекомендую уже сейчас начать учить французский, немецкий, испанский и китайский. Если у кого-то сейчас сердце пропустило удар, то не волнуйтесь – это я несерьезно. Даже если вы знаете какой-либо из иностранных языков, переводом все равно должен заняться носитель языка, так что на этом пункте в гордом слогане «я делаю видеоигру в одиночку» появляется скромное слово «почти». «Я делаю видеоигру почти в одиночку».

Помимо того, чтобы свалить локализацию игры на издателя, существует еще два варианта перевести свой продукт, не заключая лишних договоров и не позволяя никому посягнуть на процент от вашей прибыли.

Первый вариант заключается в том, чтобы обратиться в студию по локализации видеоигр. Сейчас их развелось невероятно много, но плюсы и минусы у них общие: во-первых, вам не нужно отдельно договариваться с каждым носителем языка – студия сделает это за вас. Вам же останется только поставить галочки напротив тех языков, которые вам нужны. Это, безусловно, плюс. Первый и последний.

К основному минусу относится то, что даже в самой огромной и влиятельной студии никто не гарантирует вам качественного перевода, а вы, как человек, который не говорит на всех языках мира, не сможете идентифицировать «подделку». Онлайн-переводчик поможет вам понять основной посыл текста, но определить, насколько естественно и грамотно звучит перевод, вы никогда не сможете. Мне повезло прочувствовать это на собственной шкуре, когда английское описание Reflection of Mine попало в локализаторскую контору, и в документах было решено отметить и русский язык. На выходе я получил следующее:

«Управляй двумя разными персонажами одновременно, пытаясь добраться до финиша, перемещаясь между двумя личностями сквозь множество ловушек. Каждая сторона ситуации различна для каждой личности». Следующее после этого абзаца слово «Features», после

которого перечислялись ключевые особенности игры, было переведено как «Характеристика».

Общий смысл эта малограмотная бредятина передает, но литературным текстом от нее и не пахнет. Такой же ужас, судя по отзывам иностранных игроков, творился и на других языках.

Вторым существенным минусом работы со студией-переводчиком является ценник. Он чаще всего абсолютно невменяем и устроит только тех разработчиков, которые или еще не знают про существование альтернатив, или просто сорят деньгами.

Однажды мне требовалось перевести всего сто семьдесят слов на пять языков – это снова было описание игры, только уже не моей. Я обратился в две студии, которые занимались переводами, и одна из них предложила выполнить эту непосильную работу за 270 долларов, а другая – за, черт возьми, 320 долларов. Это было в 2020 году.

Такая цена обусловлена тем, что в нее входит оплата не только работы переводчика, но и оплата работы менеджеров, аренды офиса, уборщиков и кофе-машины, рядом с которой для офисного планктона разложен бесконечный запас печенья. Человек же, делающий настоящую работу – а именно сам переводчик, – не увидит и половины затраченных на перевод средств.

Потому я решил не раскидываться деньгами, а потратить чуточку больше времени и выйти на переводчиков лично. Для поиска носителей языка отлично подходит такой ресурс, как Fiverr.com. Если переводчик в локализаторской студии безлик, неизвестен и скрыт как от ваших глаз, так и от глаз игроков, то здесь вы будете общаться лично, а на самом сайте у него будет «статус», сохранить который – в его интересах. Ответственность и заинтересованность в качественном выполнении своей работы у фрилансера будет куда выше, чем у переводчика, принимающего заказы от локализаторской студии. Это менеджерам студии придется отдуваться за некачественный перевод, а не ему. Он вас вообще не знает. И даже если у него есть какой-то к вам вопрос – ему будет проще угадать ответ, нежели связываться с вами через ужасно длинную и медлительную цепочку менеджеров, помощников и заместителей.

Потратив около часа на поиск нужных людей, я перевел сто семьдесят символов на пять языков и заплатил в общей сумме аж двадцать долларов. Самым дешевым стал французский – за него я

отдал три доллара. Жалоб от носителей языка не поступало. За час я сэкономил больше трех сотен долларов. Когда масштаб текста больше, то и экономии получится еще больше.

Третий вариант перевода своей игры является самым «независимым» из всех: вы можете надеяться, что переводчики выйдут на вас самостоятельно и предложат свои услуги за простое упоминание в титрах. Их амбициями выступит желание «прощупать кухню» разработки видеоигр, наработать портфолио или просто выразить симпатию лично вам или вашему проекту.

Вносить такой вариант в свои планы определенно не стоит, потому что появление альтруистов весьма стихийно и непредсказуемо. Далеко не факт, что судьба вас столкнет на просторах Сети с каким-нибудь жизнерадостным бразильцем, который бодро переведет весь ваш проект на португальский и не попросит ни цента. Когда он появится, воспользуйтесь этой возможностью. Но целенаправленно ждать его – дело, обреченное на провал.

Если же вы работаете с издателем и хотите скинуть перевод вашей игры на его плечи, то имейте в виду, что существуют такие сумасшедшие договоры, где часть затрат на локализацию тяжким бременем обрушивается на и без того уставшие плечи разработчика. Разумеется, так быть не должно. Издатель должен быть заинтересован в распространении вашей игры, а значит, добавление новых языков – сфера его интересов.

Локализация является очень сложной темой. Во время игры вы наверняка не раз сталкивались с ошибками в переводе. И я говорю не только о чудовищных монстрах, которые нам выдавал машинный перевод от «Фаргуса», но и о современных играх, разработчики которых добавляли русскую локализацию самостоятельно. Так, например, в игре Control от авторов гениальной Max Payne, описание одной из способностей, звучащее как «Health + 41 %» вместо «41 % к здоровью» перевели как «Урон при падении +41 %». Маленькая девочка, которая в русской версии игры Katana Zero общается как самый лютый на районе гопник, в оригинальной версии общается как обычная девчушка. Почему локализаторы решили обогатить ее лексикон отборным тюремным сленгом – сложный вопрос. Фраза, принадлежащая ей и звучавшая в оригинале как «Now split it out! Where are you hiding Leviathan?», была переведена следующим

образом: «Вываливай, собака сутулая! В воровской карман спрятал Левиофана?». Изумительно «точный» перевод, совсем не влияющий на характер персонажа и задумку авторов, правда?

Как разработчику мне бы очень хотелось свалить вину за все смысловые, стилистические и даже грамматические ошибки в текстах на локализаторские студии: подобно тому, как издатель делегирует обязанности рекламировать игру маркетинговым агентствам, он перекидывает и текст из вашей игры в другую компанию. Но, увы, обвинить абсолютно во всем локализаторские конторы я не могу: очень часто вина за чудовищный перевод лежит именно на разработчиках.

Разумеется, ребята, занимавшиеся локализацией Control, знали, как переводится слово «Health» и понимали, что это далеко не «урон от падения», и девочка из Katana Zero могла начать выражать мысли как гопник из-за отсутствия четких указаний на то, кто это вообще такая, сколько ей лет и как она выглядит. Нередко я видел в играх ошибки, когда женский персонаж начинал говорить о себе в мужском роде – фразу «I was working» переводили как «я работал», не имея, видимо, представления о том, кто эту фразу произносит.

Причины возникновения множества ошибок могли скрываться в очень плохо подготовленных для локализации материалах. Безусловно, локализаторы тоже могут делать свою работу плохо, но отвечать за чужую халатность – не в наших силах. Мы можем только попытаться минимизировать количество ошибок со своей стороны.

Чтобы увеличить шансы получить достойный перевод, нужно позаботиться о нескольких вещах еще в самом начале разработки. Если ваша игра будет хорошо развиваться – перевести ее на другие языки станет обязательным делом, и не нужно в самом начале разработки рубить на корню потенциал вашего проекта.

Во-первых, еще на стадии выбора шрифта проверьте, какие языки и символы он поддерживает. Помимо русского и английского выясните, поддерживает ли он немецкий, французский, испанский, китайский и, желательно, японский. Для каждого европейского языка есть фраза, подобная «съешь еще этих мягких французских булок, да выпей чаю» – она содержит все буквы русского алфавита, и если какой-то символ в вашем шрифте отсутствует, то вместо одной из букв вы увидите уродливый квадратик. Обязательно предоставьте

используемый шрифт и локализаторам – им как носителям языка будет гораздо проще найти плохо читаемые или отсутствующие символы.

Если же вы все еще не уверены, что внимательно проверили шрифт, то заранее позаботьтесь о том, чтобы ваша игра могла сменить его «на лету». Скорее всего, для азиатской группы языков это понадобится. И ни в коем случае ни под каким предлогом не «зашивайте» текст в текстуры! Иными словами, если вам показалось хорошей идеей сверстать текст в Photoshop, сохранить его как картинку формата. png и засунуть в игру в таком виде – подумайте о том, как тяжело вам будет что-то в нем исправить.

Я начинал делать Reflection of Mine как бесплатную мобильную игру. Изначально я хотел, чтобы в ней был один язык – английский, оттого идея с текстурами не показалась мне такой уж и глупой. Когда проект разросся и в нем появился русский, китайский, японский, немецкий, испанский и французский языки – я потратил бессчетное количество времени перерисовывая текстуры и переделывая все диалоги.

Во-вторых, не подстраивайте интерфейсы под размер используемых вами фраз – всегда должен быть запас. Некоторые фразы, например на немецком, будут втрое превосходить по длине эти же фразы на китайском. В итоге какая-нибудь надпись на немецком вылезет за границы интерфейса и будет смотреться чудовищно. Для того чтобы точно знать, не «уехал» ли текст за границы экрана или окна, в котором он располагается, у локализатора должна быть возможность без взаимодействия с разработчиком изменить в игре текст и в реальном времени пронаблюдать, как он смотрится после изменений. Ответственность за предоставление такой возможности целиком и полностью лежит на вас, потому что никто иной в коде вашей игры разобраться не сможет.

В-третьих, оформляя таблицу с текстом своей игры, обязательно добавляйте туда как можно больше контекста. Голая стена разрозненных реплик породит ошибки в духе «я работал» вместо «я работала», потому что переводчик просто не будет понимать, кто произносит эту фразу. Без понимания контекста он перепутает не только пол персонажа, но и не сможет придать фразе правильную эмоциональную окраску. Иными словами, обязательно подписывайте, кому из персонажей принадлежит какая реплика и какой текст

относится к интерфейсам. Если вы сопроводите хотя бы часть реплик скриншотами из сцен, где они произносятся, то это позволит переводчику глубже погрузиться в контекст.

Более того, подготовка материалов для перевода поможет и вам, потому что будет проще в них ориентироваться, и бережное отношение к своим текстам лишним уж точно не будет. Если вы составите еще и описания персонажей, то сами для себя же четче сформулируете особенности их характера и речи. К листу с описаниями героев можно добавить их изображения, а само описание вполне может состоять из пары абзацев про возраст героя, его пол, манеру речи и ключевые моменты биографии.

В-четвертых, стоит учитывать, что на перевод игры нужно время. Многие начинающие разработчики весьма вольно обращаются со своими текстами, воспринимая их как весьма «дешевое» наполнение игры. Ведь и правда кажется, что проще построить ключевую сцену целиком на обмене репликами, нежели прорисовывать несколько дополнительных анимаций или портретов для выразительности персонажей. Но, когда вы ознакомитесь с ценами на перевод этого текста и когда вы поймете, что переводчикам нужны месяцы на то, чтобы справиться с поставленной задачей, ваше желание забить игру графоманией начнет иссякать.

Для того чтобы изменить текст уже после локализации, вам потребуется масса времени, чтобы новую (или измененную) фразу перевести на все имеющиеся у вас языки. Без этого понимания разработчики продолжают манипуляции с текстовым наполнением игры вплоть до самого выпуска, воспринимая слова как весьма гибкий инструмент. Напротив, это самый неповоротливый элемент из всех, что есть у вас в проекте. Весь текст должен быть готов желательно хотя бы за месяц до того, как игра выйдет в свет.

С другой стороны, выпуск игры только с русским и английским языками в меню настроек не является каким-то уж совсем ужасным вариантом. Я выпускал свои игры именно так – с поддержкой всего двух языков, а остальные языки добавлял по мере их поступления.

При разработке первого проекта я принял такое решение из-за отсутствия средств на перевод. На английский мою игру перевел американец, который некоторое время жил у нас дома по программе «коучсерфинг», подразумевающей бесплатное размещение у себя

туристов с целью пообщаться, обменяться опытом и просто весело провести время. Ему понравилась игра, и он участвовал в разработке по собственной воле.

Catmaze вышла в Steam сразу с наличием китайского языка, и наплыв китайских игроков чуть не свел меня с ума – их материализовалось очень много. Научиться с ними взаимодействовать оказалось непосильной задачей: когда китаец сообщал мне о баге в игре или просто высказывал о ней свое мнение, в большинстве случаев я вообще не мог понять, какую мысль он хочет донести, потому что сообщения эти ребята присылают или прямо на китайском, или, если вам повезет, они сами прогоняют текст через google-переводчик. Просто ради экономии собственных нервов я выпустил Fearmonium снова с поддержкой только русского и английского языков. Остальные языки появлялись в нем постепенно.

И последнее, о чем стоит позаботиться, – это о наличии какого-либо чит-кода, который позволит вашим переводчикам проходить игру, не заморачиваясь с убийством боссов или решением головоломок: сборка для переводчиков должна позволять им включить, например, бессмертие или убийство монстров с одной кнопки. Уважайте чужое время. Так вы и сами получите перевод гораздо быстрее.

Текст в игре является элементом, куда более сложным и дорогим, чем вам может показаться сначала. Старайтесь использовать в игре только тот текст, который действительно необходим. Не увлекайтесь графоманией и излишествами в описании сюжета, предметов и персонажей.

29. Готовимся заявить о себе

Ничто не вводит начинающих разработчиков в такой ужас, как маркетинг. Раскрутка собственного продукта – это деятельность, которую многие из нас по степени сложности сравнивают с самой разработкой. И разработка, и маркетинг идут параллельно друг другу и требуют колоссальных затрат сил. Все догадываются, что если не потратить ресурсов на маркетинг, то о вашей игре мало кто узнает.

Программы действий по раскрутке игры ни у кого толком не сформировано, и это вводит многих из нас в ступор, провоцируя на бездействие. Но наступает ли ступор действительно из-за непонимания что делать? А может, каждый из нас все понимает, но надеется на то, что существует некий секрет, как сделать игру популярной, и известен этот секрет лишь маркетологам?

Я общался с авторами популярных игр, я наблюдал за работой маркетологов в студиях, я составлял план по раскрутке с маркетинговым агентством, которое выводило на рынок действительно крупные проекты, я и сам умудрился раскрутить три своих игры до того уровня, что они целиком окупили себя и до сих пор позволяют мне спокойно существовать в размеренном темпе работы (и это при том, что разработаны они для весьма узкого круга потребителей ввиду своих тем и жанров).

С учетом накопившегося опыта я могу с уверенностью сказать: никакого секрета у маркетологов нет. Все двигаются приблизительно по одному и тому же пути, спотыкаясь на одинаковых преградах. Не существует секретного способа, используя который вы сделаете свою игру популярной одним щелчком пальцев.

Я упоминал, что издатель занимается маркетингом, вкладывая в продукт деньги. Но что делать нам, многоруким разработчикам-одиночкам? Неужели тоже нанимать маркетолога? Это было бы странно, с учетом того, что мы уже объединили в себе кучу профессий, которые ничуть не проще маркетинга. Так что в копилку навыков нам придется добавлять еще один.

С одной стороны, мы находимся даже в более выгодных условиях, чем люди, переложившие проблему маркетинга на чужие плечи: как-то

раз я помогал раскручивать игру, разработчики которой были чертовски сильно заинтересованы в том, чтобы им самим было здорово в нее играть. Это абсолютно верное решение с их стороны, и оно помогало довести разработку до конца, но, когда пришло время представлять игру общественности, авторы столкнулись с тотальной незаинтересованностью игроков в их продукте. Редко проскакивающие комментаторы в жизнерадостных постах под игрой прямо указывали на ее видимые недочеты. Как разработчик я понимал, что лучший способ собрать аудиторию заключался в том, чтобы поменять некоторые незначительные аспекты игры и вложить больше ресурсов в общий дизайн. Но я не выполнял роль разработчика и никак не мог повлиять на внешний вид игры – я должен был помогать маркетологу работать с тем, что мы имели.

При разработке игры в одиночку мне было гораздо проще реагировать на то или иное замечание от сообщества. Я легко мог что-то убрать или, напротив, добавить в том случае, если это не противоречило моей оригинальной задумке.

Так, например, в моей пошаговой головоломке Reflection of Mine именно «по вине» первых игроков появился счетчик ходов.

По моей первоначальной идее игроку нужно было просто пройти уровень, не задумываясь о количестве совершенных им действий, но в итоге в верхней части экрана появилась цифра, указывающая минимальное количество шагов, за которое можно добраться до финиша на данном этапе. Игроки стремились достигнуть этого результата или даже превзойти его – я специально к минимальному значению ходов прибавлял парочку, чтобы тем самым создать у самых находчивых игроков ощущение, что они обманули и переиграли создателя этой игры. Как игрок я не являюсь большим фанатом соревнований, но путем добавления соревновательного элемента я расширил целевую аудиторию и увеличил степень удержания игроков в Reflection of Mine.

Я не считаю, что это сильно повредило игре. Идея добавить такой элемент едва ли возникла бы в моей голове, но и сопротивляться его появлению, цепляясь за свои принципы, я не стал – не так уж для меня и важен этот счетчик. Он просто есть.

Создавая Fearmonium, на первых парах я держал в голове мысль, а будет ли эта сцена круто смотреться в трейлере? Некоторые анимации

я создавал специально для того, чтобы они классно выглядели в маркетинговых материалах. Они, безусловно, приятно дополняли и саму игру, но главной их целью было быстро «насытить» картинку общей атмосферой мрачного веселья с примесью безумия и эротики – я говорю сейчас конкретно про анимацию телепорта путем запрыгивания в ванну к соблазнительному антигерою Леди Депрессии. Я придумал эту анимацию с целью создать яркий маркетинговый материал.

Fearmonium создавал свою атмосферу путем пересечения множества элементов, и для того, чтобы прочувствовать происходящее, игроку нужно было именно сидеть и играть несколько минут. Несколько минут внимания игрока – это роскошь в маркетинге. У маркетолога есть несколько секунд, чтобы привлечь внимание к своему продукту.

Подобным образом ведется и реклама игр-ужастиков: все мы знаем, что основную часть времени в хоррорах мы решаем головоломки, исследуем окрестности и упорно ищем способ попасть в новые локации. Монстры выпрыгивают на нас достаточно редко, потому что в хорошем ужастике важны отнюдь не те моменты, когда мы видим, как монстр тянет к нам свою страшную рожу из очередного незакрытого шкафа. Самые важные сцены в ужастиках – это когда мы вглядываемся в темноту, неуверенно светим туда фонариком и понятия не имеем, есть там кто-нибудь или нет. Хороший ужастик – это когда мы догадываемся, что монстр стоит за углом, а не смотрим ему прямо в глаза.

Но, несмотря на это, в рекламе любого хоррора обязательно будет момент с кричащими мордами разных чудовищ, потому что промоматериалы – это «спрессованное» содержание игры. Иной раз приходится специально создавать моменты, способные хорошо и быстро продемонстрировать какой-либо элемент, который очень сложно показать без плотного вмешательства маркетолога в разработку игры.

AAA-игры делаются с огромным количеством маркетологов в команде, что, к слову, очень хорошо влияет на продажу разработанных с таким подходом продуктов. Не нужно человеку особо долго объяснять, чем ему может понравиться очередной Assassin’s Creed и что собой представляет данная игра – достаточно ему показать скриншоты. Иначе дела обстоят с оригинальным Dwarf Fortress,

визуальный стиль которого особо не вызывает интереса, и убедить игрока заценить эту игру – уже куда сложнее *(рис. 13)*.

Планируйте начальные стадии разработки так, чтобы у вас как можно быстрее набралось материалов, которые раскрывают суть вашей игры. При их демонстрации вы быстро сможете понять, с распростертыми объятиями или, напротив, со скептицизмом встречают ваш проект. Не имеет никакого смысла рисовать для главного героя все анимации сразу: быть может, публике чудовищно не понравится его дизайн, или, что еще хуже, зрители найдут исключительное сходство вашего героя с персонажем из какого-нибудь другого проекта, в который вы, возможно, даже и не играли.

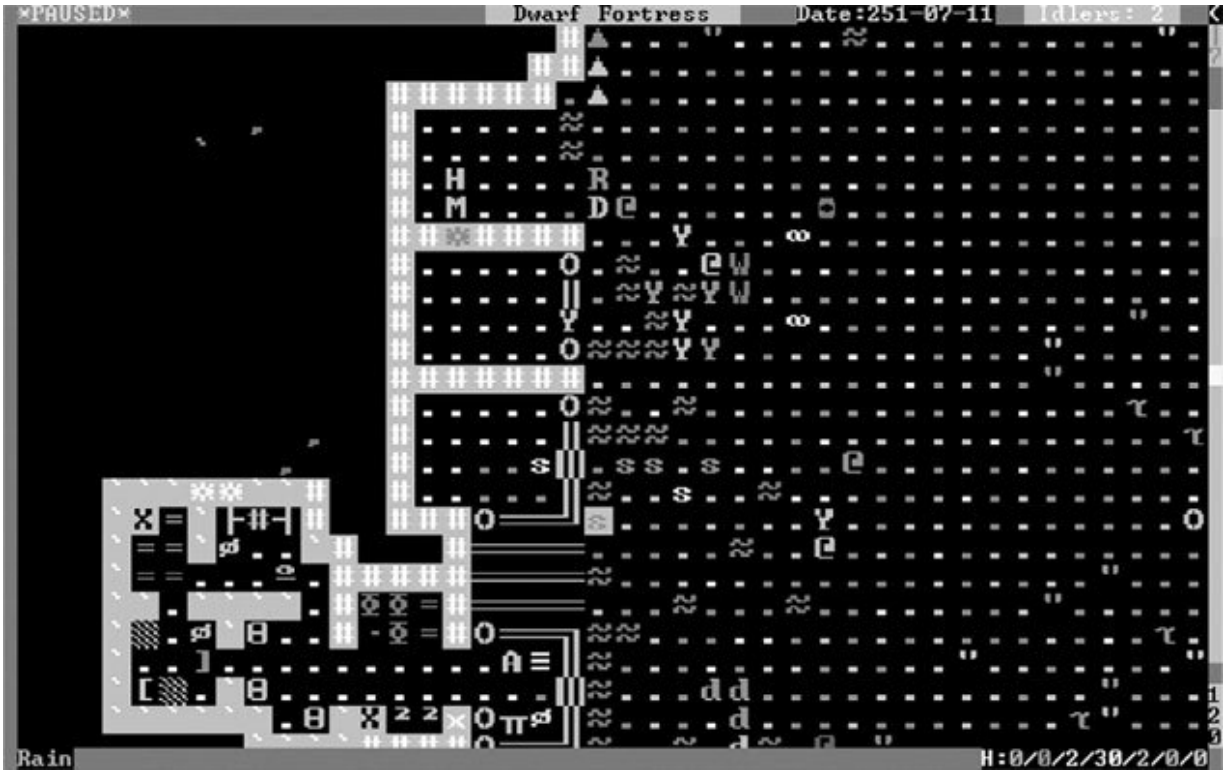

Рис. 13. ***Dwarf Fortress, 2006 год***

В том, насколько активно вы будете подстраиваться под настроение общественности, кроется ответ на вопрос, коммерцией ли мы тут занимаемся или творчеством. Кто-то считает мои проекты невероятно авторскими, а кто-то, напротив, обвиняет меня в создании сугубо

коммерческих продуктов, пытающихся походить на игры, в которые я даже не играл.

Мнения о вашем подходе всегда будут диаметрально противоположными. Для кого-то и спортивные ААА-игры – искусство. Комментарии в адрес того, насколько ваш продукт коммерческий, не так важны, как ваше внутреннее ощущение и ваши собственные причины заниматься разработкой игр.

Определиться с тем, ради чего вы создаете игровые миры – ради прибыли или ради самовыражения, – стоит в самом начале пути. Нужно это для того, чтобы знать, как реагировать на критику и в каком направлении развивать игру. Если вы выбираете прибыль, то анализировать комментарии игроков нужно внимательнее, не стесняясь вносить изменения в свой проект под веянием моды и настроя игроков.

В случае же, когда в приоритете находится ваше собственное видение игры, вы поймете, что вам абсолютно не обязательно прислушиваться к отзывам в духе «игра выглядит убого» или пытаться разобраться, что там сейчас популярно на рынке. Все решения будут приниматься исходя из «мне так нравится, и все тут». Я, например, сторонник второго варианта. Я не пытаюсь угодить всем и не стремлюсь создавать коммерчески успешные продукты.

Мы должны выбирать позиции в сложных вопросах, чтобы исходить из принятых убеждений в решении встающих на нашем пути проблем. Если вы выбрали «не париться насчет высказываний в адрес моей графики», то вы всегда будете сразу знать, что вам делать при прочтении очередного комментария, критикующего выбранный вами стиль.

30. Наводим марафет в Steam

В современном мире у вашей независимой игры будет одна самая важная страница, которая должна привлечь и игроков, и представителей СМИ. Я говорю про страницу игры в Steam. Ее неописуемо важно оформить правильно и красиво, потому что, как бы вы ни старались, у вас вряд ли самостоятельно получится продемонстрировать ваш продукт бóльшему количеству людей, нежели это сделает площадка от Valve. В случае когда игра попадает в важнейший раздел «популярных новинок», переходы на эту страницу исчисляются миллионами *(рис. 14)*.

Если вы серьезно нацелены выпустить игру в Steam, то страницу с игрой стоит завести как минимум за сто восемьдесят дней до выпуска игры. Разумеется, сама игра может еще находиться в разработке все это время – вам просто нужно иметь страницу, где, во-первых, собрана информация о вашей игре, а во-вторых, есть кнопка «добавить в желаемое». Так пользователи смогут «подписаться» на ваш проект, и им автоматически придет на почту письмо, когда ваш продукт выйдет в свет.

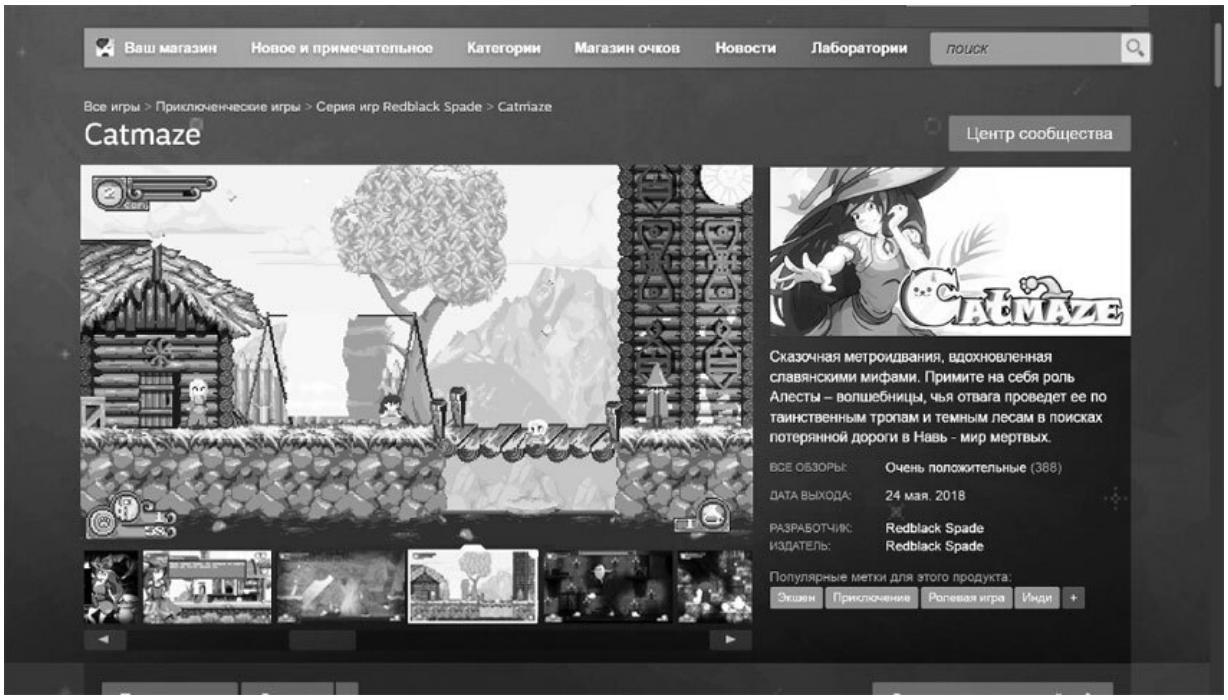

Рис. 14. ***Страница игры Catmaze в магазине Steam***

Самое первое, о чем стоит подумать, – это название вашей игры. В последнее время я предпочитаю скрещивать два уже существующих слова, чтобы создать новое – Cat + maze = Catmaze, или же использовать латинские суффиксы: fear + monium = Fearmonium. Этот способ хорош тем, что позволяет получить лаконичное и уникальное название, а также использует слова, которые игроки частенько используют в поиске и могут случайно напороться на мои игры.

Этот трюк весьма распространен: **Dead** Cells, Hollow **Knight**, **Dark**est **Dungeon**, **Fear**monium, **Cat**maze.

Лаконичность важна из-за того, что, чем меньше в вашем названии букв, тем меньше вероятность, что игрок, запомнивший название на слух, ошибется в его написании во время поиска игры. Поиск по неуклюжему запросу «Firmonum», например, не приведет игрока на страницу с моей игрой.

Уникальность названия важна из-за того, что поисковые системы и алгоритмы могут попросту «утопить» вашу игру в миллионе похожих результатов. Специальные плашки в Google, YouTube и Twitch с информацией о вашем проекте формируются автоматически. Если название оригинальное и встречается преимущественно в статьях

именно про вашу игру, то плашки появятся быстрее. Если же игра названа одним каким-нибудь популярным словом, например, Hype, то вам придется конкурировать аж с 1,280,000,000 результатов, которые выдает Google по этому запросу. Вероятность, что поисковые системы приведут пользователей на страницу именно с вашей игрой, будет крайне мала.

Перед тем как определиться с названием, обязательно вбейте его в поисковой строке в Google. Чем меньше будет результатов – тем лучше. Когда я вводил запрос «Fearmonium» до анонса самой игры, поисковик выдавал мне около одной тысячи результатов. Я понимал, что потеряться в таком малом количестве информации будет сложно, а это значит, что название я выбрал правильное.

Второй важнейший элемент – это «капсульные изображения». Steam попросит вас загрузить множество промокартинок разных размеров для того, чтобы демонстрировать их в различных частях магазина, но я настойчиво рекомендую не видоизменять основную суть вашего главного промоизображения и лишь незначительно модифицировать композицию для того, чтобы все важные элементы влезли в требуемый размер картинки. На каждой картинке для промо используйте один и тот же логотип, одного и того же персонажа, один и тот же фон, одну и ту же палитру.

Важно работать над узнаваемостью. Игрок вполне может проигнорировать попавшуюся ему на глаза «капсулу» вашей игры один раз, но, когда в других разделах магазина Steam он увидит одну и ту же картинку еще десять раз, он явно обратит на нее внимание. Она будет попадаться ему на главной странице в разделе новинок, в разделе со скидками, в библиотеке, в рекомендациях и т. д. Игроки доверяют рекомендациям платформы, и если Steam активно пытается обратить их внимание на игру, то рано или поздно это получится. Не усложняйте сами себе жизнь, рисуя или заказывая различные промо-изображения для всего многообразия разделов в Steam.

Не относитесь к Steam как к некому «рыночному развалу», куда пользователь приходит, чтобы взять конкретный товар в куче мусора и уйти с ним восвояси. Алгоритмы Steam – это интересная и сложная система, в создании которой талантливые маркетологи использовали все свои знания о человеческом мозге.

Если вы зарегистрируетесь в этой системе, купите и поиграете хотя бы в несколько игр, то главная страница Steam начнет подстраиваться под вас, предлагая вам продукты, похожие на те, в которые вы играли. В ходе эволюции инстинкт выбирать «похожее» развился в человеческих особях до предела. Если в первобытные времена дикарю попадались два вида ягод, один из которых он видел первый раз в жизни, а второй уже когда-то ел и насытился, то он без сомнения выбирал второй сорт ягод, чтобы увеличить свои шансы на выживание. Такой образ мышления остался и у нас. Когда Steam говорит нам, что «вот эта игра, в которую ты не играл, очень похожа на ту игру, в которую ты играл», это звучит как «ты уже ел такие ягоды. Ты знаешь, что этот продукт безопасен».

Когда в нашей группе в VK «ИНДИ ИГРЫ | INDIE GAMES», посвященной независимым играм, я делаю пост о каком-нибудь новом проекте, в комментариях с огромной долей вероятности появится человек, который заявит, что продемонстрированная мной игра похожа на какую-то другую игру. В процессе поиска сходства нового продукта со старым продуктом, архивирующая часть его мозга «решает» в какую ассоциативную цепочку воткнуть только что усвоенные образы.

Если мы увидим новую черно-белую 2D-игру, то архивирующий мозг с огромной долей вероятности начнет по-старчески бубнить следующее: «Игра черно-белая? Ага… мы когда-то играли в Limbo. Limbo – черно-белый. Так что давай напишем в комментарии, что эта игра похожа на Limbo». Этот процесс – естественен. Он обеспечил выживание нашего вида. Но Steam его очень умело использует.

К сожалению, часть процессов, которые Steam будет использовать для продвижения нашей игры, останутся для нас неизведанными, а сама площадка предоставляет не так уж много метрик для анализа аудитории. Я настоятельно рекомендую добавить на страницу с игрой Google Analytics – это сервис, который позволяет собирать данные о посетителях вашей страницы. Так вы сможете узнать их точное количество; увидеть, в каких регионах ваша игра наиболее интересна, а также собирать информацию о том, сколько людей находятся на странице с вашей игрой в данный момент. Все, что требуется от вас, – это зарегистрироваться в сервисе Google Analytics, найти там

специальный код, скопировать его и вставить в особую графу в настройках вашей страницы в Steam.

Но кое-что из встроенных инструментов нам серьезно пригодится. В настройках страницы Steam можно отыскать всего три метрики: количество показов, коэффициент переходов и количество посещений. Такой параметр, как «коэффициент переходов», отображает именно то, настолько выбранное вами капсульное изображение нравится игрокам. Адекватная цифра в этом случае должна быть больше 15 %. Это значение будет свидетельствовать о том, что из сотни игроков, увидевших ваше промоизображение, 15 перешли на страницу. Если показатель меньше – меняйте капсульные изображения, иначе коммерческий успех вашего продукта окажется под большим вопросом.

При смене изображения я рекомендую основной упор сделать на смену цветовой палитры – так, например, простое перекрашивание гор и облаков из агрессивно красного оттенка в более приглушенные пурпурные тона на капсульном изображении Fearmonium увеличил коэффициент переходов с 9 до 18 %.

Значение такого параметра, как «показы», отображает то, сколько раз промоизображение вашей игры демонстрировалось игрокам в Steam. Алгоритмы этой платформы самостоятельно решают, где и сколько раз продемонстрировать капсульное изображение вашей игры, но пускать все на самотек не стоит: у нас есть силы повлиять на этот результат с помощью системы «тегов» – специальных меток для игры, которая помогает алгоритмам определить, что вообще собой представляет ваш продукт: стратегия это или интерактивное кино? Файтинг или головоломка?

Инструмент tag wizard позволяет вам выставить 20 меток к вашей игре. Содержание меток и, что немаловажно, их порядок определят то, каким игрокам и в каких разделах ваша игра будет высвечиваться. Если вы выставили себе метку «стратегия» и «темное фэнтези», то тем игрокам, которые играли в игры с такими же метками, с бóльшей вероятностью выдадут рекомендацию поиграть и в вашу игру.

Если вы считаете, что у вас мало показов, – меняйте метки. Если у вас маленький «коэффициент переходов» – меняйте капсульные изображения. Если меньше 10 % посетивших

страницу людей добавило игру в желаемое – меняйте описание игры.

Steam сразу предлагает вам оценить «правильность» выбора меток, давая вам взглянуть на игры, которые он определил как «похожие» на вашу. Чаще всего там будет список из очень популярных игр, но не стоит надеяться, что вы сможете попасть к ним на страницы в раздел «похожие игры». Ваша игра попадет на чужие страницы и будет активно продвигаться площадкой, только если станет такой же популярной: в этом разделе игры ранжируются не только по меткам, но и по количеству игроков. Иными словами, если основной меткой игры является «метроидвания», то в «похожих играх» на ее странице вы увидите десять самых популярных метроидваний.

Выходит, «чтобы быть популярным – нужно быть популярным». Увы, но так и есть. Steam будет куда активнее продвигать те игры, которые и без того хорошо продаются, используя такую человеческую черту, как конформизм: проще соблазнить человека играть в то, во что играют «все», потому что, как я уже упоминал, люди – это стадные животные, и потерять свое место в стае является одним из наших основных страхов. Если вся стая играет в эту игру, то стоит ею хотя бы поинтересоваться, верно?

Но такой расклад не делает систему tag wizard бесполезной, и метки все еще очень важны. Помимо очевидного раздела «похожие игры» метки определяют появление игры в рекомендациях отдельных пользователей. Главная страница Steam формируется для каждого игрока особенным образом, и вы никогда не будете до конца понимать, как и откуда к вам переходят покупатели. Ясно одно: чем больше вы их заманите, тем больше их будет появляться в дальнейшем.

Чтобы лучше понять, кому Steam демонстрирует вашу игру, я рекомендую воспользоваться двумя сервисами. Первый – **steamlikes.com** – поможет разобраться, на страницах каких продуктов она отображается в разделе «похожие игры». С ростом популярности вашего детища таких страниц будет становиться больше, так что не стоит расстраиваться, увидев в самом начале пути очень скромный список малопонятных игр. Значение в 15–20 игр уже хорошее. Второй сервис – **steampeek.hu**, поможет определить, играющим в какие игры Steam будет рекомендовать ваш проект. Стоит иметь в виду, что

страницы на этих сервисах обновляются не моментально и после того, как вы добавили или поменяли метки, стоит подождать несколько часов, а иной раз и дней.

Таким образом, существуют три вещи, сигнализирующие о том, что с вашими метками что-то не так и было бы неплохо их поменять:

• полное отсутствие похожих игр на steamlikes или же наличие там всего одного-двух проектов;

• подобная ситуация со steampeek, который в случае неправильной работы с метками на запрос с названием вашей игры выдаст среди «похожих» проекты, не имеющие к вашему никакого отношения;

• третьим параметром правильно выставленных меток послужит количество «показов» вашей игры, и на этом мы остановимся подробнее.

Ответить, какая цифра в графе «количество показов» является нормальной, невозможно: иной раз демонстрация игры сотне игроков, увлеченных жанром, в котором вы работаете, приведет к бóльшему количеству переходов, в то время как десятки тысяч просмотров от любителей AAA-проектов никаким образом не конвертируется в переходы.

Много лет назад Steam гарантировал один миллион нецелевых показов в день выхода игры. Судя по статистике от моих коллег, толка от подобного рода продвижения было меньше, чем от демонстрации игры двадцати тысячам целевых игроков.

Самым главным мерилом успеха в продвижении вашей игры станет количество «виш-листов». Когда игрок переходит на страницу с игрой, которая еще только находится в разработке, вместо кнопки «купить» он увидит кнопку «добавить в желаемое». Когда игра выйдет, такой игрок получит на свою электронную почту письмо в духе «игра из вашего списка желаемого теперь продается». Именно эти игроки и должны обеспечить успех самого важного события в жизненном цикле вашей игры – ее выпуска.

Когда в Steam выходит очередная игра (а их выходит несколько тысяч в год), она не попадает на главную страницу магазина. Чтобы найти игру, придется листать эту страницу до раздела «популярные новинки», в нем нажать кнопку «новинки», на открывшейся странице снова выбрать закладку «новинки» и перейти там по ссылке «все новинки». Вы можете себе представить, какое малое количество

игроков заходит в этот темный и неприметный раздел. Это – лимб. И если ваша игра оказалась только там, то ее заметит не очень-то внушительное количество игроков.

Оттого основной целью по выпуску игры в Steam является попадание в раздел «популярных новинок» на главной странице. Количество копий, которые нужно продать в первые часы выпуска, все время варьируется, и точного числа назвать невозможно: все будет зависеть от игр, вышедших на одной неделе с вашей. Десятка лучших игр попадает в популярные новинки. Первые продажи должны обеспечить люди, добавившие игру в список желаемого и получившие письма на электронный ящик, а когда вы окажетесь на главной странице Steam, тогда подтянутся и тысячи других игроков, услышавших о вашем проекте впервые и обративших на него внимание по той причине, что он позиционируется как «популярный».

Весьма очевидно, что если на одной неделе с вами выйдет Half-Life 3, очередной Battlefield, новый Call of Duty и семь ремастеров приквела к сиквелу спиноффа Resident Evil, то никакого места в «популярных новинках» для вашей небольшой игрушки не останется – вам будет очень тяжело переплюнуть продажи этих суровых и раскрученных монстров. Но если рядом с вами выходили только совсем уж ничем непримечательные независимые игры, сил в раскрутку которых особо никто не вкладывал, то у вас есть все шансы заявить о себе на главной странице Steam.

Свою статистику по количеству людей, добавивших игру в «список желаемого», можно увидеть в финансовой информации вашего аккаунта Steam. Обновляется она раз в сутки, где-то около 14:00 по московскому времени, так что не расстраивайтесь, когда после публикации страницы у вас некоторое время будет отображаться депрессивная цифра «ноль» в разделе Wishlists.

Поскольку вероятность попасть в «популярные новинки» зависит от даты выпуска как вашего проекта, так и игр от ваших конкурентов, невозможно назвать, сколько точно нужно иметь «вишей» для сто процентного попадания в желанный раздел. За первые двое суток игру купит приблизительно 3–5 % от людей, добавивших ваш проект в список желаемого. Если этого хватит для того, чтобы попасть в популярные новинки, то еще столько же человек купит игру сразу – это будут новые игроки. Где-то через неделю соотношение людей,

добавивших игру в список желаемого и людей, купивших игру, должно составить 1: 10 – каждый десятый игрок, добавивший ваш проект в список желаемого, в итоге его приобретет.

Если вам тяжело понять, насколько популярны ваши «конкуренты», выпускающие свои проекты с вами в один день, то можно воспользоваться следующим фокусом: на странице с любой игрой внизу есть кнопка «найти группы сообщества». Если в найденной группе больше людей, чем в такой же группе у вас, это значит, что данная игра популярнее вашей и можно подумать о том, чтобы сместить свою дату выпуска на какой-нибудь другой день. Эти группы создаются автоматически. При оформлении страницы с игрой вас заставят оформить и их, так что не пропустите: иначе администрация Steam просто не позволит вам опубликовать страницу с игрой.

На вопрос, сколько нужно «вишей», чтобы попасть в «популярные новинки», нет однозначного ответа, есть лишь рекомендации. Моя рекомендация – двадцать тысяч «вишей». Чтобы собрать такое количество заинтересованных в игре человек, стоит завести страницу в Steam как можно раньше. Желательно за год до выпуска самой игры.

Приблизительная статистика, позволяющая вам рассчитать прибыль еще до выхода игры, выглядит так: 10 % игроков, добавивших ваш проект в список желаемого, его все-таки рано или поздно приобретут. В ходе самой первой же недели из списка желаемого игру «выкупят» где-то 3–4 %. Если этого будет достаточно, чтобы ваш проект попал в популярные новинки, то еще столько же человек приобретет его, просто наткнувшись на новинку на главной странице Steam. Доход, полученный на первой неделе и умноженный на пять, составит ваш доход за год в том случае, если вы регулярно будете участвовать в распродажах. Все эти цифры очень приблизительные, ибо очень многое зависит, разумеется, от качества самого продукта.

Для оформления страницы вам не нужна готовая игра, так что не имеет смысла доделывать проект, оформлять страницу, а потом ждать. «Виши» пополняются параллельно с разработкой. Для страницы вам потребуется хотя бы четыре скриншота и крайне рекомендуется наличие видеоролика, но это не обязательно.

Скриншоты должны отображать разнообразные ситуации из вашей игры и отвечать на самый главный вопрос, который возникнет у

игрока: а что это за жанр и что тут надо делать? Потому не стоит убирать со скриншотов интерфейсы – многие игроки понимают, к какому жанру отнести игру, именно по ним. Если на момент публикации у вас нарисованы, например, всего две локации, то обязательно чередуйте скриншоты: нужно сделать так, чтобы цветовая палитра и композиция у каждого последующего скриншота максимально отличались от каждого предыдущего.

Видеоролик не обязан представлять собой двухминутное видео, раскрывающее все механики вашей игры. Для первого видео хватит и 30–40 секунд, самыми главными из которых будут первые пять: именно столько времени в нашем суетливом современном мире у вас есть, чтобы привлечь внимание зрителя. Если зрителя ничего не зацепит в первые пять секунд, то вероятность, что он досмотрит видеоролик, снижается. Оттого ни в коем случае не нужно забивать первые секунды трейлера всплывающими из темноты логотипами с названием вашей никому неизвестной «студии» и абсолютно никому неизвестной игры. Вы не FromSoftware, которой достаточно показать логотип Dark Souls XXVIII, чтобы у игроков отвисла челюсть. Первые секунды вашего трейлера – это «трейлер к трейлеру», когда вы демонстрируете самое интересное, что есть в вашем проекте.

Немаловажен и «питч» вашей игры – ее короткое описание, которое располагается под капсульным изображением на странице с игрой. Я наблюдаю два вида таких описаний, и к первому из них относится «понятное, техническое». Например, у Dead Cells оно звучит так: «Dead Cells – это экшн-платформер в жанре Rogue-lite и Metroidvania. Вас ждет огромный, постоянно меняющийся замок… Если, конечно, вы сможете победить тех, кто встанет у вас на пути, в 2D-схватках в стиле Souls-lite. Без сохранений. Убивайте, умирайте, учитесь и пробуйте снова».

Такое описание прекрасно по двум причинам. Во-первых, оно использует множество терминов, знакомых только опытным игрокам и характеризующих игру как продукт со сложными для новичков механиками.

Отсечь часть аудитории – это тоже важная маркетинговая работа. Если бы Dead Cells продавалась бы под видом «гиперказуальной игры», то те, кто купился бы на столь обманчивую рекламу, в итоге остались бы недовольны и завалили бы игру грозными отзывами.

Текущее же описание Dead Cells отсекает неопытных игроков и оставляет только тех, кто готов к трудностям и уже имеет необходимый для понимания Dead Cells игровой опыт.

Во-вторых, в этом описании куча глаголов, которые мало того, что легко и быстро читаются, так еще и прекрасно описывают то, что в Dead Cells придется делать – убивать, умирать, учиться и повторять все это снова. Игроку важно понимать, чем ему предстоит заниматься в игре, и именно ответ на вопрос, что тут надо делать, он будет искать на странице с вашим проектом. Если вы игроку ответа не дадите – игру не купят. Если вы его обманете, пообещав розовых пони в игре про убийство зомби, то он оставит негативный отзыв и попросит вернуть ему деньги.

> **Используйте в кратком описании вашей игры больше глаголов. Не углубляйтесь в описание сюжета, а лучше всего – обозначьте игровой цикл вашего проекта. Игрок должен понимать, чем ему предстоит заниматься, если он купит то, что вы сделали.**

Второй вид описаний я бы назвал «Что-за-черт-описание». Вместо того чтобы объяснить игроку, что происходит в игре и что ему предстоит делать, такое описание стремится только запутать беднягу. Такими описаниями блещет студия Playdead, описывающая свою игру Inside как «Гонимый и одинокий, мальчик оказывается в самом центре мрачного проекта». Такое описание не рассказывает ничего ни о жанровой принадлежности игры, ни о ее механиках, ни о ее содержании. В дуэте с загадочным промоизображением оно лишь подталкивает игрока к тому, чтобы углубиться в поиск ответов на вопрос: а что это за игра?

Раскрыть игроку все карты и особенности или же, напротив, привлечь его внимание «что-за-черт-описанием» – решать вам, все зависит от самого проекта и от вашего воображения. Помните, что описание, заглавное изображение, скриншоты, метки и видеоролики всегда можно поменять в том случае, если вы не довольны метриками.

Вам кажется, что у вас мало показов, и вас не устраивает, какие игры вы видите на сервисах steamlikes и steampeek? Меняйте метки и смотрите, стало ли лучше. Вас не устраивает количество переходов?

Меняйте заглавное изображение и, опять же, смотрите на статистику. Вам не нравится, что даже при наличии большого количества переходов мало кто добавляет игру в список желаемого? Займитесь оформлением страницы, ведь помимо скриншотов и видео Steam предлагает вам добавить развернутое описание вашей игры.

К чтению развернутого описания прибегнет не такое большое количество игроков, как к просмотру хедера или скриншотов. Данная графа предназначена для самых докучных, и имейте в виду – чем больше там спрятано текста, тем меньше вероятность, что игрок дойдет до конца. Наполнение данного раздела должно окончательно отвечать на вопросы:

• Что нужно делать в вашей игре?

• Почему игрок должен купить именно вашу игру, вместо того чтобы уделить внимание проекту конкурента?

Куда больше, чем текст, скажут анимированные изображения: их можно вставлять в описание, зная один хитрый трюк. Расширение анимированного изображения – это. gif, и ваш файл с промоанимацией будет называться, например, animation.gif. Сам сервис Steam не позволит вам загрузить файл с подобным именем – он будет просить вас использовать. png или. jpeg – расширения для неанимированных картинок. Но если переименовать animation.gif в animation.png, то хоть на вашем компьютере этот файл и перестанет открываться как надо, при загрузке его в Steam на выходе вы увидите анимированную картинку.

Для создания анимированных материалов можно использовать один из двух методов. Первый метод заключается в записи экрана с помощью, например, программы fraps. Тогда вы получите видеофайл, который можно отредактировать и конвертировать в gif, скажем, на бесплатном онлайн-сервисе ezgif.com.

Второй способ слегка попроще – можно использовать бесплатную программку LICEcap, которая позволит захватить выбранную вами часть экрана, указать желаемый размер и частоту кадров. Чтобы определиться с тем, какие параметры FPS и разрешения применить для сохранения gif, я рекомендую побродить по площадке Steam и найти в описаниях игр анимированные изображения, качество и внешний вид которых понравились лично вам. Сохранив эти gif-файлы у себя на компьютере, вы сможете подглядеть их свойства и использовать точно

такие же параметры при создании собственных анимированных изображений.

В идеальном мире анимированные промоматериалы должны отразить игровой цикл вашего проекта. Если вы делаете ужастик, и цикл заключается в «заходим в дом – решаем головоломку – бегаем от монстра», то три анимированных картинки в полной мере смогут отразить суть вашей игры.

Текст в этом разделе позволит вам уже более развернуто ответить игроку на вопрос, в чем отличительные особенности этой игры, но чтобы ваш ответ был доходчив и понятен, не используйте стену текста. Человек, который ищет новые игры и суетливо поддается стимулам к поисковой активности, которые и привели его к вам на страницу, крайне возбужден и нетерпелив. Ему некогда пробиваться через стену графомании.

31. Выходим на сцену

Когда страница в Steam будет опубликована, не стоит полагаться только на загадочные алгоритмы и надеяться, что дальнейшее бездействие обеспечит вас достаточным количеством игроков. Стоит понимать еще одну особенность алгоритмов Steam, которая заключается в следующем: чем больше посетителей приходит на страницу с вашей игрой со сторонних ресурсов, тем значительнее становится ваша видимость и внутри Steam. Алгоритмы регистрируют заинтересованность в вашей игре, и Steam выдает вам больше просмотров внутри собственной системы.

Методов заявить о себе – множество, но смиритесь с тем, что чаще всего вам предстоит кричать в пустоту. Рано или поздно вы до кого-то докричитесь, но перед этим вам предстоит сделать множество постов, рядом с которыми будет стоять лишь одна отметка «мне нравится» от вашего хорошего друга, вы напишете множество писем, на которые никто не ответит, вы опубликуете несколько статей, которые исчезнут в дремучем болоте другой информации.

Это нормально. Крик в пустоту – это основная работа любого маркетолога в самом начале пути. Не стоит рассчитывать на то, что при первом размещении информации о вашем проекте вы соберете огромную базу фанатов. Скорее всего, напротив, абсолютно никто никак не отреагирует. Ваша задача – продолжать кричать. При должном усердии вы дозоветесь кого-нибудь.

«Кричать» нужно несколькими способами. Во-первых, разумеется, собирать сообщество вокруг своего проекта в одной из социальных сетей: это может быть VK, Twitter, Telegram и т. д. Какая из платформ вам наиболее привычна и удобна – с нее и начинайте. Подглядите, как ведут свои сообщества другие разработчики, деятельность которых вам симпатична. У нас не так много форматов для постов, и чаще всего мы пишем что-то в духе «вот недавно сделали новую анимацию» и прикладываем анимированную картинку, или же «написал код для красивых частиц», а рядом размещаем симпатичный скриншот с частицами.

Для набора аудитории в большинстве социальных сетей лучше всего работают репосты и ретвиты: вы договариваетесь с другим разработчиком о том, чтобы он в своем сообществе разместил информацию о вашей игре. Иногда это делается за деньги, иногда на условиях какого-то бартера, иногда за обещание оказать аналогичную услугу в далеком будущем: выбирайте тот вариант, который вам подойдет. Где искать коллег-разработчиков, мы обсуждали в предыдущих главах, а сейчас мы придумали, как можно сделать такое знакомство обоюдовыгодным.

В Twitter не стоит забывать про хэштеги, выбрать которые можно из приведенной ниже таблицы, а также обращать внимание на время, в которое вы делаете пост. Если вы не встретились вообще ни с какой активностью в виде комментариев, лайков и ретвитов, то нет никакого смысла оставлять этот пост висеть мертвым грузом на вашем аккаунте. Запомните, сколько сейчас времени и больше не делайте постов в этот период. Подождите минут 15–20, удалите пост и разместите его опять в другое время.

Тег	Потенциал видимости
#screenshotsaturday	15,5775
#pixelart	9,7397
#metroidvania	9,5588
#gameart	6,8611
#castlevania	4,4123
#madewithunity	4,1564
#platformergame	3,4101
#indiedevhout	2,7131
#sidescroller	2,5392

#gamedesign	2,5234
#gamedevelopment	2,258
#rpg	2,1993
#platformer	1,9164
#wip	1,8449
#dev	1,8113
#indiedev	1,7395
#indiegames	1,6896
#gamedev	1,6279
#indiegame	1,591
#indiegamedev	1,4507
#unity	1,4154
#2d	1,2712
#retrogame	1,2686
#nintendoswitch	1,2371
#game	1,0534
#retrogaming	1,0458
#steam	1,0404
#retrogames	1,002
#pixel	0,9131
#nintendo	0,8847

#screenshot	0,8815
#actionplatformer	0,8778
#famicom	0,8065
#nes	0,8032
#gameplay	0,7626
#gamer	0,7404
#videogame	0,641
#videogaming	0,5616
#games	0,5532
#videogames	0,5382
#mario	0,4902
#gaming	0,4161
#switch	0,301
#trailer	0,2374
#trailer	0,2135
#creating	0,1848
#indie	0,1552
#level	0,1354
#marketing	0,1284
#gameidea	0,0826
#video	0,0126

Самой темной лошадкой для юных маркетологов является сайт «reddit», представляющий собой огромный ресурс, поделенный на так называемые «сабреддиты», посвященные абсолютно разным темам. Они представляют собой отдельные сообщества, например, про кино,

рыбалку, еду, фотографию, но нас интересуют, разумеется, те из них, которые имеют отношение к играм. Наша задача – найти в хаосе разных тем разделы, в которых информация о нашей игре может оказаться уместной. Если мы делаем шутер – ищем сабреддиты, посвященные шутерам; если платформер – нам нужны разделы, связанные с платформерами; если мы используем в своем проекте pixel-art – к нашим услугам сообщества, относящиеся к этой замечательной форме изобразительного искусства.

Для того чтобы все ваши посты на reddit не ушли в пустоту, а отобразились в лентах пользователей, подписанных на сабреддиты, где вы размещаете информацию о вашей игре, я рекомендую позаботиться о своей «карме». Когда вам ставят «плюсик» за пост или комментарий – ваша карма растет, и, чем она больше, тем выше видимость у ваших постов.

Я особенно рекомендую две вещи: разогнать значение вашей «кармы» на комментариях хотя бы до трех сотен перед тем, как начать выкладывать информацию о вашей игре, и читать правила, всегда читать правила каждого сабреддита! Они везде разные, и их нарушение может привести к понижению кармы из-за того, что пользователи вас закидают «минусами» или же просто модераторы сабреддита потрут ваши посты.

Размещение постов в социальных сетях напоминает пальбу из пистолета-пулемета «Узи», которым вы не умеете пользоваться, и поэтому все выстрелы разлетаются в разные стороны. Но есть ресурсы, размещение информации на которых схоже с обращением со снайперской винтовкой: писать туда нужно редко, но метко.

Я говорю про такие ресурсы, как DTF, GameDeveloper (бывш. Gamasutra), indieDB, StopGame, Gamin и Pikabu. Наиболее уместным форматом для этих интернет-изданий станут большие и полезные статьи с интересными историями или с описанием необычных решений, принятых вами в разработке. Темы статей могут быть самыми разными – главное, чтобы материал был полезен пользователю, развлекал его или учил чему-то новому. Написание такого материала требует много времени, поэтому относитесь к этой работе максимально внимательно и наберитесь вдохновения от других авторов: разработчики выкладывают что-либо на эти ресурсы

довольно часто, и проанализировать, что сейчас интересно читающей аудитории, не составит труда.

Спектр возможных тем для статьи зависит от вашей конкретной деятельности, и общей рекомендации здесь дать невозможно: вам нужно самим решать, какой элемент в проекте заслуживает наибольшего внимания.

Ваши знакомства с другими авторами игр помогут своевременно получать информацию о различных конкурсах независимых разработчиков, проводимых на просторах сети. Их много, они разные, и иной раз они проходят лишь единожды, оттого списка подобных мероприятий я привести не могу – все слишком быстро меняется. Но по Сети гуляют обновляемые таблицы, и ссылка на одну из них расположена на *Рис. 15*.

Даже если вы не рассчитываете на победу и видите, что ваши конкуренты чертовски опытны и сильны, это не должно послужить поводом опустить руки и не подавать заявки. Я как-то участвовал в конкурсе от Игромании, и несмотря на то что я занял там всего пятое место, информация об игре появилась на этом огромном ресурсе, а качество моей игры прокомментировали художники из Blizzard. Назвать такое событие «бессмысленным» язык не повернется.

До журналистов из Игромании и других подобных ресурсов всегда есть способ «достучаться» и через почту. Письма журналистам и блогерам – это отдельная наука, тонкости которой я попытаюсь раскрыть в следующей главе, пока скажу коротко: перед тем как кому-то писать, нужно составить «пресс-кит».

Пресс-кит представляет собой набор промоматериалов, которые помогут журналисту быстрее понять суть вашей игры и подготовить про нее материал. Наличие пресс-кита – это очень хороший тон, тем более что почти все необходимое у вас появилось при оформлении страницы в Steam. Сейчас нужно просто красиво это оформить.

https://docs.google.com/spreadsheets/d/1NGseGNHv6Tth5e_yuRWzeVczQkzqXXGF4k16IsvyiTE/edit#gid=0
Рис. 15.

Итак, создаем папку, которую ни в коем случае нельзя называть просто Press-Kit. Когда я писал про игры, таких папок у меня в «загрузках» было неимоверно много, и я понять не мог, к какой игре какая папка относится. Обязательно добавляйте к имени папки название игры. Папка, которая называется Fearmonium Press-Kit, тут же даст понять, что же там у нее внутри.

А внутри будет папка Screenshots, куда вы добавите скриншоты из вашей игры, а также папка GIFs, куда пойдут анимированные картинки (если они, конечно, уже есть), и еще папка с логотипом игры и различными промоизображениями (будет великолепно, если некоторые из изображений будут предоставлены в виде. png файла без фона – так журналист сможет с помощью них симпатичнее оформить статью или видеоролик), и, наконец, несколько текстовых файлов – один с описанием игры, другой с информацией о вас самих, а третий документ должен содержать все ссылки – на страницу в магазин, на трейлер, на ваши социальные сети и любые другие контакты, которые сочтете нужными. Не надо вкладывать никаких видеофайлов в пресс-кит – журналисту нужна ссылка на YouTube, а не файл. mp4 или. avi, который ему придется куда-то загружать, чтобы поделиться роликом со своими читателей.

Всю эту папку вы загружаете на Dropbox, Google drive или же на собственный домен, если таковой имеется, и готовитесь к крайне ответственной работе: рассылке писем.

Ваш пресс-кит – это лицо вашего проекта. Однажды ко мне обратились как к издателю, предоставив мне только название игры. На мою резонную просьбу скинуть пресс-кит мне посоветовали просто «загуглить» игру самостоятельно. Разумеется, отвечать на такое

хамство я не стал – никто не будет работать с теми, кто не бережет чужое время.

Пресс-кит состоит из логотипов компании, логотипов игры, разноформатных промоизображений, скриншотов, анимированных изображений (если есть), а также из текстового файла со ссылкой на видео на YouTube, файла с описанием игры и файла с ссылками на все ваши социальные сети и контакты.

В современном мире у множества деятелей Интернета уже собрана своя аудитория, которая ждет от них обзоров или рассказов о новых играх. Некоторые блогеры с сотнями тысяч подписчиков смогут рассказать о вашем продукте аудитории, которую вам придется целенаправленно собирать несколько лет.

Привлечь их внимание – это работа, выполняемая в несколько этапов. Сначала нужно создать список журналистов и блогеров, которым ваша игра может показаться интересной. Я рекомендую не откладывать это дело на потом, а прямо в ходе разработки отмечать где-нибудь в блокнотике попавшихся на глаза интернет-деятелей, чье творчество вполне совместимо с вашей игрой.

Для экономии времени можно присмотреться к сервисам вроде keymailer.co – он позволяет разослать ключи заинтересованным в вашей игре блогерам автоматически, без использования почтовых ящиков и переписки.

Лично я искал ребят, делающих обзоры и прохождения игр, схожих с моими по жанру. Я сохранял информацию о них в табличке из четырех колонок: имя, количество подписчиков, ссылка на канал или паблик и электронная почта. Заполнение последней колонки обеспечивало меня самой сильной головной болью: не все популярные интернет-деятели размещают свои адреса в свободном доступе, иной раз нужно внимательно поискать заветную информацию в их профилях на различных ресурсах. Другая проблема заключается в том, что на YouTube блогеры прячут свои адреса в специальный раздел в окне «about», и от вас потребуется нажать на кнопку «показать мне адрес». Google будет считывать, сколько раз вы это сделали, и уже на десятый

начнет вас мучать утомительной капчей со светофорами и лодками, а потом и вовсе запретит просмотр адресов.

Проблема решается весьма просто: надо завести еще несколько аккаунтов Google и постоянно их менять: ограничение в демонстрацию 10–15 почтовых ящиков действует не на ваш IP-адрес, а именно на аккаунт. Я не рекомендую выбирать в качестве новых псевдонимов набор букв, потому что эти адреса нам понадобятся чуть позже.

Содержание письма зависит от того, кому вы пишете и какую информацию вы хотите до него донести, но логика построения текста всегда будет одинаковой. Начать стоит с приветствия, содержащего имя или псевдоним самого блогера – так вы сразу дадите понять, что ваше письмо не является частью массовой рассылки (заниматься которой абсолютно бессмысленно). После этого представьтесь сами и затем дайте короткое и звучное описание вашей игры. Обязательно стоит прикрепить анимированное промоизображение, демонстрирующее основную механику проекта. Помните: ваша задача заключается не только в привлечении внимания, но и в демонстрации уважения ко времени человека, которому вы пишете: пробиваться через полноразмерные скриншоты, переходить по ссылкам или сразу смотреть видео – это утомительно и долго, а вот анимированное изображение моментально предоставит нужную информацию об игре и поможет журналисту понять, интересен ему этот проект или нет.

Дальнейшее содержание письма зависит от стадии готовности вашей игры и непосредственно от ваших целей. Если уже готова демоверсия или же игра близка к выпуску и вы с помощью сервиса Steam сгенерировали для нее специальные одноразовые коды, позволяющие пользователям получить доступ к продукту прямо сейчас, то прикрепляйте ссылку на демоверсию или предоставляйте журналисту его личный код. Тогда он сможет сделать на вашу игру обзор или рассказать про демоверсию. Если же нужно только информационное освещение и у вас не имеется ничего, во что можно поиграть, то в письме сразу после чуть более развернутого описания можно закругляться. Используйте вежливое прощание, за которым обязательно последует ссылка на пресс-кит, страницу в Steam и трейлер.

Написание таких писем требует немало времени, но благо в нашем распоряжении есть черновики и таймеры, так что я умудрился примерно в один день отправить три сотни подобных писем перед выпуском своей игры. На их написание мне потребовалось около двух недель, хотя я уже и был научен предыдущим опытом и использовал несколько почтовых ящиков.

Дело еще и в том, что если вы не получите ответа на 50 отправленных вами писем, то сервисы Google начнут направлять всю исходящую от вас корреспонденцию в папку «спам», что, разумеется, очень-очень плохо. Но тут и приходят на помощь наши почтовые адреса, что были созданы ради просмотра контактов у различных блогеров: поверьте, журналисту будет все равно, с какого адреса отправлено письмо. Если оно не попало в спам, внутри был найден ключ от игры, а почтовый адрес не напоминает удар лбом о клавиатуру, то вопросов ни у кого не возникнет.

Разумеется, вы можете настроить рассылку с собственного домена, и вам это удастся без всякого труда, если вы уже оперируете такими аббревиатурами, как DKIM, SPF и DMARC, но в данном случае можно не усложнять себе жизнь и обойтись без приобретения знаний столь узкой квалификации.

Блогер или журналист вряд ли будет отвечать на письмо, даже если и правда решит снять ролик или провести стрим по вашей игре. А уж если игра его не заинтересует, то тем более не рассчитывайте на ответ. Все дело в том, что часть разработчиков реагирует на отказ крайне болезненно, начиная хамить в ответ или требуя от журналиста разъяснений по поводу того, что же не так с его игрой. Подобные разговоры ни к чему, кроме траты времени блогера, не приводят. У вас не получится словами убедить человека в том, что игра, которая ему не нравится, все-таки хорошая. Такое бесцеремонное поведение и провоцирует журналистов не отвечать. Каким бы умным и порядочным вы бы ни были, у вас этого на лбу не написано, и журналисту невдомек, насколько адекватный человек ему пишет. Шанс нарваться на невоспитанного психопата – крайне велик, так что блогер вполне вероятно предпочтет никого не провоцировать и просто молча делать свою работу.

Молчание может расстроить, а проделанная по рассылке писем работа может показаться бессмысленной, когда из сотни блогеров

лишь десять сделают какой-либо материал по вашей игре. Но даже если сто отправленных писем привели к тому, что о вашей игре рассказало лишь десять человек, у вас нет никаких причин расстраиваться – без этой сотни писем не появилось бы вообще никакого материала. Десять – лучше, чем ноль.

Я стараюсь не забывать, что для меня разработка игры – это создание готового к распространению продукта, а без должного маркетинга назвать игру «готовой к распространению» невозможно. Потому маркетинг – это такая же часть разработки, как рисование и программирование. Без маркетинга вы не донесете своих идей до тех людей, чью жизнь они могут поменять.

32. Не свихнись

Мои игры не расположились на вершине хит-парадов. Они не являются продуктами, нацеленными на широкую аудиторию и созданными для коммерческого успеха. Через придуманных мной персонажей я изливал свою душу, а в их речах формулировал свои идеи. Разумеется, мои работы остались незамеченными и непонятыми очень многими, и в этом нет никакой проблемы, потому что делать игры «для всех» – не то, чем мне хотелось бы заниматься, и не то, чем я рекомендую заниматься вам.

Игр «для всех» очень много. Мне же хочется создавать то, что метит буквально в десяток сердец, и даже одно попадание в цель будет значить, что отведенное на разработку время не было потрачено зря.

Небольшие независимые игры, оказавшись в сердце игрока, проникают куда глубже, чем коммерческие продукты: неотесанные черты наших игр должны угодить самым сокровенным предпочтениям игрока; тем его интересам, которые делают этого человека особенным и отличающимся от остальных. Мы не привлекаем его внимание только за счет моды, агрессивного маркетинга или принадлежности к крупной франшизе.

То, какие моральные невзгоды приходится преодолевать во время разработки, делает наши проекты более личными: мы не работаем за зарплату, мы не находимся в зоне комфорта; создавая игру, мы жертвуем временем, которое могли бы провести за развлечениями или за общением с родными. Чтобы эта жертва не казалась чрезвычайно мучительной, мы учимся получать от процесса создания видеоигры удовольствие. Потому что в удовольствии и кроется ответ на вопросы, которые могут все еще оставаться у читателей – откуда взять время на разработку и как не свихнуться по пути?

Действительно, у начинающих разработчиков часто стоит вопрос о том, где найти время на создание собственной игры и на приобретение необходимых для этого навыков. Ответ на этот вопрос кроется не в том, чтобы пожертвовать одной из сфер своей жизни в угоду новой мечте – пожалуйста, не нужно так делать. Секрет кроется в том, что какими бы деловыми мы ни были, у нас всегда находится время на то,

от чего мы получаем удовольствие. Просто иной раз мы этого не осознаем.

Однажды, выслушивая очередные жалобы от знакомого мне разработчика на то, что он ничего не успевает, я попросил его показать статистику пользования смартфоном. Когда я увидел колоссальные «пять часов в день» – ответ на вопрос, почему он ничего не успевает, показался мне очевидным. Не удивительно, что ему нравится разбрасываться своим временем – психологи годами работали над тем, чтобы использование мобильных приложений и социальных сетей захватывало и поглощало нас.

Секрет «успевания» кроется не в том, чтобы дисциплинированно ограничить себя в получении радости от бесполезной деятельности, а в том, чтобы от разработки видеоигр получать не меньшее удовольствие, чем от игры в них. Мне нравится делать игры сильнее, чем сидеть в социальных сетях или смотреть фильмы. По этой причине в свободные минутки я почти всегда выбираю созидание, а все эти минутки складываются в итоге в часы, дни, недели и месяцы.

Посмотрите на себя глазами ребенка. Своими же глазами из далекого детства. Если бы «семилетний я» узнал, что «тридцатилетний я» зарабатывает на жизнь созданием игр, то этот мелкий пацан, у которого в кармане лежит несколько «кэпсов» с персонажами из Mortal Kombat, раскрыл бы рот от изумления и выронил бы из рук контроллер от Sega Genesis. Ему было бы абсолютно плевать на то, насколько мои игры успешны. Ему не важно, насколько у меня много шансов «выстрелить» своим следующим проектом. Для него главное – чтобы я продолжал делать игры. Его восхищает, что я могу довести до выпуска любой из своих проектов и научиться вещам, которые он считает невозможными.

Кто из нас не хочет быть тем, кем его видит настолько восхищенный ребенок?

Мотивация, так необходимая всем нам, не является вопросом силы воли и лежит больше в эмоционально-чувственном спектре. Нужна ли вам мотивация, чтобы проводить время за любимым занятием? Чтобы проводить время с любимыми людьми? Будете ли вы читать книгу о том, как найти силы для лежания на диване и как научиться получать радость от солнечных дней?

Когда у вас опускаются руки, а мечты и амбиции превращаются в ваших врагов, которые мучают совесть и высасывают силы, наступает самое время чтобы проанализировать контекст работы и лишний раз напомнить себе о том, почему вы вообще собрались делать игры.

Получаемая информация формирует ваше мышление, поэтому возвращаться к тому, что вас вдохновило в самом начале пути, – всегда хорошая идея. Помните, что отказ от потребления мусорной, поверхностной и суетливой информации не может не оказать благоприятного влияния на творческую сферу вашей жизни.

Невозможно раздавать универсальные советы о том, как выйти из лабиринта прокрастинации. Можно лишь подсказать направление. Я даю компас, а не карту. Пусть эго каждого из нас требует успехов на поприще разработки видеоигр сильнее и громче – никогда не давайте натиску невзгод и трудностей убедить себя в том, что вам «и без выпущенной видеоигры нормально живется». Если уж начали созидать или хотя бы интересоваться – это значит, что потребность, которую вы стремитесь удовлетворить разработкой видеоигры, все-таки существует. Усталость, выгорание и упадок сил не отнимут подобной потребности. Эти демоны лишь сдуют размер вашего эго, уместят его в крошечное офисное кресло на пластиковых колесиках и будут катить вас все дальше от того человека, которым вы хотели когда-то стать.

Основной инструмент в разработке видеоигр – это не движки, программы или приспособления вроде планшета или компьютера. Основной инструмент – это ваш мозг. Никакие знания мира и навыки мастера не помогут сделать игру, если вы не осознали особенностей своего мышления.

Любые двери открываются тогда, когда ваши глубинные убеждения формируют для них свой собственный ключ. Ключ к разработке видеоигр теперь у вас есть.

ВИДЕОИГРЫ
ГЛУБОКОЕ ПОГРУЖЕНИЕ

СЛАВА ГРИС

СДЕЛАЙ ВИДЕОИГРУ ОДИН

И НЕ СВИХНИСЬ!