Всем привет. Меня зовут Владимир Алямкин я техлид в Пушкин Studio — это студия внутри My Games то есть текущего Mail.ru Group. Сегодня у меня на удивление не технический доклад он скорее посвящен некоторым общим вопросам связанным с идентификацией себя вообще как разработчика в мире Unreal. Откуда этот вопрос возник, будучи техлидом я достаточно часто собеседую людей помогаю другим студиям собеседовать тех, кто приходит. И очень часто мы сталкиваемся с одними и теми же проблемами при подборе кадров при сортировке резюме при личном собеседовании. И причём сталкиваемся с этой проблемой с обоих сторон как со стороны будущего работодателя так и со стороны соискателя. На некоторые такие вопросы я и хочу ответить сегодня в моём достаточно коротком докладе в силу нашего тайминга.

Начну с первого вопроса. Кто такой вообще анрильщик? То есть мы работаем с Unreal Engine. Мы что-то на нем делаем. Когда можно говорить я анрильщик или можно ли говорить так? Или вообще нельзя говорить. Что это вообще за персонаж такой? На мой взгляд использование вот этого слова анрильщик это чисто идентификация. Я работаю с Unreal Engine. Это абсолютно ничего не говорит о скиле человека. О его направлении о его опыте о его стремлениях и более того о текущем его состоянии и о том с чем он работает. По сути все заявления о том, что я там умею в Unreal это фикция. Unreal достаточно широкая штука нельзя уметь во всё. и знание какого-то инструментария без его поддержки в отсутствии базы это тоже всё очень временное и разовое. это не та история которую ты будешь проецировать вперёд не та история которая на самом деле работает.

Начну с некоторого такого провокационного слайда. Кто точно не анрильщик? Человек который делает только блюпринты ну не опытный анрильщик можно так сказать. Есть такие задачи опять же читайте между строк. Я не говорю что если вы умеете только блюпринты всё дорога закрыта. Нет если вы работаете там вы 3D художник да. То есть вы создаёте модельки вы их как то оптимизируете. Импортити. Экспортити. Это тоже это очень частный опыт работы. то есть если вы только модельки делаете без настройки шейдинга без оценки и детали рендеринга. Если вы гейм дизайнер, который что-то прототипирует быстрое это опять же не погружает вас в Unreal даже если вы 5 лет с этим проработали. К сожалению самый такой жирный юзкейс это виарщики. Ко мне на собеседование приходило очень много виарщиков. С большинством не о чем разговаривать мы делаем игры мобильные, десктопные. Мы пишем логику. К сожалению, такой большой бум VR создал к тому, что человек умеет что-то настраивать, но не может работать дальше. Именно на такие кейсы и рассчитан мой доклад. То есть это не значит, что VR плохо это не значит, что все, кто работает с VR-ом ничего не знают нет ни в коем случае есть огромный спрос на грамотных людей.

Дальше. Кто такой анрильщик? На мой взгляд можно разделить две составляющие это либо технари либо артисты. Всем остальным чаще всего не требуется знать вообще движок. Ни дизайнерам. Ни там дизайнерам интерфейса. Всё это вторично под что ты работаешь. Какая разница Unity, Unreal, CryEngine это настолько детали нюансы что они не являются основ образуемыми. Но если вы технарь Вы должны разбираться, как ни странно, как работает звук. Как работает Game Framework. Из чего он состоит. Какие базовые куски у него присутствуют. Как вообще работает рендер. Что такое рендер в игровом движке в конце концов. То есть большая проблема бывает как? Что-то делает человек абсолютно не знает как устроен рендер. Как установить рендер для игр? Как это работает в случае c arwizard? И не задаётся этими вопросами. На уровне базовом это знать надо. Именно базовом. О чём расскажу немножко дальше. Сеть тоже хорошая штука. Опять же если вы не знаете сеть. Если вы к этому не стремитесь вообще открещиваетесь это тоже нормальное явление. Не надо это воспринимать как полный список знаний это лишь примеры. Тоже самое касается артистов, когда артист может быть считаться анрильщиком, когда артист может считаться эффективным, когда он понимает нюансы работы под конкретный движок. То есть как этот движок обрабатывает нормали. Как он импортит. Как устроены оси в конце концов. Как происходит обработка тех или иных шейдеров. То есть по сути анрильщик это тот человек который знает нюансы движка применительно к своей профессии и используют их во благо или знает как их обойти. Более того настоящий анрильщик живёт аптудейт. Меня очень смущает люди, которые сидят на версии, например недавно 4.11 и говорят, что они идут в ногу со временем. Сейчас 4.23 превью 3 и даже мои знакомые музыканты уже изучили что там такое интересное, а когда программисты сидят на старых версиях или даже сами не посмотрят вызывает вопрос.

Очень часто моя специфика я всё-таки технарь и в первую очередь наверное ближе к программистам. Встает вопрос нужны ли плюсы при игровой разработке и мой первый слайд блюпринтера не являются анрильщиками. В чём проблема Unreal? С одной стороны, плюсы — это достаточно сложный язык со своими нюансами, требующий изучения требующий подхода им не так просто начать пользоваться очень легко выстрелить себе в ногу любимое выражение. С другой стороны и на блюпринтах можно написать огромное количество игр. Достаточно сказать там ну например на фортнайте там 70% условно кода это блюпринты. Я прекрасным образом знаю много проектов у которых огромная часть блюпринтового кода и это тоже нормально. Для оптимизации можно сказать включили нативизацию. Всё работает. Всё хорошо. Но что такое плюсы в контексте Unreal это не просто знание там каких-то инструкций. Нет это умение с ними работать то есть что самое важное в программирование это алгоритмы структура данных это общий набор компетенций с которым вы работаете который вы используете дальше. Если вы не знаете, что такое контейнеры как с ними работать как не знаю, чем отличается указатель от ссылки в плюсах то там логика, которую вы будете писать она изначально будет не техническая. Она изначально не будет учитывать огромное количество нюансов и просто десятилетие накопленного опыта именно в программировании. Десятилетие человеческого опыта доступного глобально. То есть вы можете сказать нет мне это не нужно но если вы откинули как бы сами плюсы как язык а вы при этом технарь и програмист вы откинули этот опыт. Вы пишите свои велосипеды вы используете как бы очень большие блоки данных. То есть вы строите дом этажами. Игры так не работают. То есть если в прототипах это нормально. Продаткшн не работает так. Вы строите дом допустим не из кирпичей но из отдельных там стен даже не комнатами работа блюпринтами это работа ну до уровня комнаты которым вы застраиваете этаж. Работа с кодом с построением — это более глубокая работа вы можете это не использовать, но вы должны знать и чувствовать, как с этим работать вот и всё. Поэтому когда говорят что плюсы для Unreal не нужны ну да что на них писать там набежит память. Всё очень просто никаких там шаблонов. Они конечно есть но безусловно читать плюсовой код Unreal гораздо проще чем модуль C++17 и то что обсуждают на сипипишных конференциях. Этого нет в Unreal, но это есть в головах тех людей кто пишут Unreal. И чтобы это понимать, чтобы этим пользоваться неплохо было бы это изучить. Соответственно я очень часто говорил о таком понятии нужна база, нужна база, нужна база, что же это такое база. В первую очередь база для технаря да всё-таки для технаря это алгебра и геометрия школы. Как ни странно в большей части кода который пишется для большей части игр это просто математика 8-11 класса. Не знаю написать кастомную физику для танка наш рабочий проект это математика 10-11 класса и Википедия. Там нет никакой магии там нет никаких сверхзнаний там не требуется разработать систему. Вы пишите не симуляторы там, где важны там все нюансы и 50 переменных различных там или даже сотня. Одно дело вы пишите симулятор например какой-нибудь там мотоцикла так чтобы все нюансы резины отрабатывали. Это одна ситуация. Возможно, там требуется уже курс расширенной математики Вузовский да. Вы пишите игровые логику Вы должны знать, что такое Линейная прогрессия. Что такое геометрическая прогрессия. Вы должны понимать, как работает интерполяция в конце концов. Если вы этого не знаете если вы не умеете работать с векторами на уровне посчитать расстояние между предметами. Вы не можете писать игровую логику. Более того вы не сможете этот опыт экстраполировать дальше то есть это то на чём строится всё на чём строится определение смотрите вы на предмет или не смотрите. Включить сейчас звук или выключить. Все эти алгоритмы это база это школьная база математики. Это просто надо знать. Когда мне приходят люди и не могут посчитать простейшие такие вещи как я не знаю там перемножение векторов. Точнее объяснить зачем это надо. Не надо на листочке это выписывать это вызывает вопросы. Более того алгоритмы. Почему важны алгоритмы? Game Dev.Простейшая задача. Сортировка mavow по значению. У вас есть данные. У вас есть имя игрока. Есть очки игрока. Выведите мне пожалуйста отсортировано по значению эту таблицу. Турнирная таблица задача есть в большинстве игр. Как ни странно огромное количество опять же огромное количество людей приходящих на собеседование не знает как это сделать. Не то что даже грубый какой-то вариант да не то что там оптимизированный вариант. Не знают как в принципе это сделать и утыкаются в косяки. Ну было бы смешно если бы не было так грустно. Также база именно в Game Dev это знание того как работают игровые движки. Как работают игровые движки на уровне объектов. Например взять там Unity. Взять Unreal. Взять CryEngine. У них в принципе одна и та же архитектура. Сейчас не берём там как ECS и подобные всякие вещи. Мы сейчас говорим о обычной ооп. Работу с памятью расскажу немножко позже.

Что базой не является? Накопленный опыт работы на блюпринтах или знания функции. То есть не так давно приходит ко мне человек и рассказывает: Вот я там 5 лет знаю там те то функции он начинает. Говорю ну у тебя их нету. И все и ступор у человека, то есть если ты знаешь просто набор функций это ничего тебе не обещает. Ты умеешь просто что-то складывать это не база — это просто накопленный опыт он конечно полезен безусловно, но не является основообразующим. Тоже самое настройка классов и объектов на движке. Вы сказали да блин я знаю как тут настраивать вот у меня есть коректор мумант. Я знаю как настроить перемещение персонажа вот у меня синхронизируется всё летает всё работает вот у него маршруты прокладывается. Как это устроено внутри? База — это то, что эти объекты делают внутри. Это не то что они делают снаружи для вас это написали вы просто пользователь. Это незнание базы это знание прикладного API. Это не даёт ничего кроме как бы опять же опыта применения куда посмотреть. Ну это как уметь в редакторе запекать свет то есть да там знать огромное количество окошечек. Я знаю художников которые знают больше окошечек в редакторе чем я. Это нормально абсолютно я никогда в них не лез. Ну я знаю как это работает внутри и могу им рассказать. Ещё интересным примером является некий stackoverflow programming это бездумное копирование чужих решений. Ну то есть вот попытка, например написать сортировку капипейстов. Или я даже не знаю попытка полностью скопировать уроки по Unreal Engine и спрашивать почему вот здесь вот не работает, а там просто человек каст лишний поставил или версии движка изменилась. А всё вот также как было, так и скопирована ладно это не база.

Что я рекомендую смотреть на эту тему? Есть прекрасная книжка вот такая вот синяя она есть в бумажном варианте любому технарю и программисту просто вот стоит её порешать. Это олимпиадные задачи по программированию для школьников. Если вы там взрослый человек считаете себя программистом с опытом или что-то. Прежде чем там заявлять я програмист, я знаю там Unreal, я знаю как маршруты прокладывать порешайте оттуда задачки. Их можно легко предложить на Unreal то есть они касаются там без разницы какого языка они немножко другого. Они учат думать. Они учат выстраивать эти данные. Всё остальное касается просто понимания как работает трехмерная графика. Игры на текущий момент это трехмерная графика. Что такое полигоны, что такое вертится, что такое дракон, что такое шейдера, текстуры. Я всё перечислил из такого как это всё взаимосвязано? Как работает GPU в конце концов? Просто задать вопросы. Зачем вот, например на тиренг системах новых к nvideo они сделали такую вещь как мешлец. Ну наверное это специфические знания но знание общего порядка зачем это происходит они хорошо развивают базу. Вы растете в ширь. Вы узнаете зачем это всё. Какие ограничения существуют на текущий момент.

Джентльменский набор называется, к чему вообще был доклад у меня статья написана ещё год назад. Это первая часть вторая часть выйдет на следующей неделе. Там можно просто найти для себя списочек вопросов на которые стоит ответить если вы готовы называть тебя хотя бы медлом. Это не означает что все надо вещи надо знать в деталях но уметь на эти вопросы отвечать стоит. Опять же многим людям это статья помогла я с ними общаюсь я достаточно активно участвую в сообществе поэтому рекомендую посмотреть. Я не стал я хотел сделать доклад который презентует вторую часть только вот в виде моего доклада но потом понял что лучше это иметь в тексте чем в видео. Видео и докладе касаясь более общих тем куда смотреть.

Как раз таки памятка HR. Каждый кто приходит к нам либо сетевой либо сеньор. Окей. Хорошо. У нас вообще отсутствуют джуниоры там по резюме. Не надо этому верить у всех опыт со всем возможным. Составлять резюме научились за несколько лет. Всё хорошо. Резюме все красивые. Вообще ноль претензий к большинству. Спрашивают статью с прошлого слайда. Да это можно сказать вот ты ее сам написал и рассказываешь. Нет серьезно расспросите пусть человек хотя бы это посмотрит то есть очень многие кто в комьюнити сидят эту статью знают. Не только я кудо то тыкаю скажем так я только первый раз куда-то в доклад тыкнул. Человек посмотрит вопросы задаст это как показывают полезно. И Спросите если есть в зале то есть примеры боевых задач чтобы человек подготовился к собеседованию. Будет гораздо приятнее спрашивать. Не знаю почему такая практика когда приходит собесетоваться человек на чистые плюсы. Программист не знаю maps me и приходят к нам собеседоваться программист на плюсовика анрильщик хочется спросить а плюсы ты как-то подготовил? Спросил? Собеседование всё-таки.

Касательно грейдов. Достаточно шустро пройду по таким вопросам как себя позиционировать? Джум, Мидл, Сеньор. Там всё на самом деле система грейдов может быть широкой но это один из подходов. То есть есть джум, мидл, сеньор. Есть некие такие звери как лид дизайнеры. Чем они наглядно отличаются на самом деле вот картинка показывает. Вопрос что ты умеешь делать с граблями, когда некоторые перед тобой лежат.

Joom классика жанра человек начинает разбираться с чем-то. То есть это человек которому можно дать задачу он ее сделает под присмотром. Он ее сделает неоптимально. На самом деле сложно делегировать задачу людям, которые только учатся. Ты знаешь что результат который будет теперь придётся до пинать. Тебе придётся его править. Тебе придется объяснять зачем. Это нормальное явление когда то есть когда все мы когда-то учились. Все мы когда-то делали странные вещи но основное что можно выделить joom не может ответить на вопрос зачем и почему здесь сделано так. То есть ну вот в другом коде не в своём. В своём он скажет почему потому что вот так вот он считает лучше. Свой код все объяснят. Почему в чужом коде зачем и почему в коде сделано так joom не может ответить на этот вопрос. Это как раз качественный переход, но надо помнить, что, если человек вообще какую-то чушь говорит это не joom ну это вообще такое тоже бывает. Joom учится.

Middle основная Рабочая лошадка любой разработки. Просто основная рабочая лошадка это человек который делает задачи. Ты даёшь человеку задачу он её делает. Ты смотришь. Задача может быть сделана неоптимально но чем условно более продвинутый middke тем более оптимальная задача. Но это именно работа над задачами. Человек знает как он это делает задачу. Он может аргументировать. Он может как-то посмотреть чужой код и сказать почему он сделан так. Это самая широкая категория на самом деле она достаточно размазана. Потому что с одной стороны это только что вышедший там joom. С другой стороны, это может быть уже почти синьор. Но именно этот человек который просто берёт и делает. Он делает. Он справляется со своими задачами. Он их решает.

Следующая категория это синьор. Задача этой категории специалистов это оценивать функционал в общем. То есть это такой человек не делает задачу которую ему поставили. То есть не таск он делает. Он решает задачу в целом. Есть вещи которые в движке например просто не решаются напрямую но нет такого функционала или он противоречит. Но часто постановка откуда задача идет. Не от других отделов это как бы межконфессиональная. Геймдизайнеры попросили сделать то-то. Художники попросили сделать то-то. Ответ Джуна я вообще не знаю как это сделать. Ответ мидла это невозможно. Ответ сеньора скажет, а зачем вам это? То есть это попытка разобраться в задачи и решить её другими способами. Поэтому это просто человек который смотрит в общем. Ну от частности не обязательно знать этот вопрос куда посмотреть дальше то есть вот этот пункт очень важен. Сеньор не знает всего это ключевое его аспект. Есть лиды. Они вообще ну они не про знания. Это про команду и про ответственность. Программер может быть мидлом. он просто руководит командой разработки. Руководит так что сеньоры условно делают квадривиум. Мидлы работают над задачами. Джуниоры учатся.

Ну и категории вакансии. Когда увидите любую вакансию Вы должны по ее тексту и позиционированию понять чего от вас хотят. Либо ищут человека кто расскажет как делать ищут сеньора. Тогда ну извините ребята будут искать тех кто им расскажет как делать а не собеседовать вас какие-то технические вещи. Это вот реально поиск сеньора. Кто будет делать? Это вопрос поисками мидлов. Ну то есть, когда ищу человека с опытом, но которому скажут, что делать. И третье это кто будет закрывать объем. То есть это ну есть куча задач маленьких чтобы учить джуниоров.

Дальше я немножко проскачу два слайда очень быстро. Про тестовые задания. Ребята сейчас на рынке перегрев. Тестовые задания встречаются везде. Я сталкиваюсь иногда с тем, что тестовые не нужны. Нет они нужны. Они нужны чтобы понимать ну как бы производить некий первичный отбор. Очень много людей на рынке и очень огромный спрос и всё это устаканивается. Через несколько лет это станет ненужным. Пока это нужно относитесь кто соискатель к этому нормально. Это всё обсуждаемо всегда.

И прекрасный ответ как знать все. Ответ никак. Ты можешь просто стремиться. Ты просто делаешь свои проекты. Ты делаешь пэдпроекты. Экспериментируешь над чем-то. Ты смотришь вперед. Тебе никто не расскажет, как и что делать если ты не сможешь это сделать сам, то есть ты просто стремишься и развиваешься это надо запомнить, как мантру следовать этому никто ни в одном рабочем проекте не сделает так чтобы, вы развивались шире горизонтально это исключение из правил вы всегда будете расти в каком-то узком ключе чтобы расти в широком ключе. Ну участвуйте в сообществе задавайте вопросы и в общем-то всё будет хорошо.

И последний слайд на этом я заканчиваю. Какие опасности есть на пути? Первое. Быть лучшим среди худших ребята. Если вы пришли в команду, в которой Вы лучше, но при этом вы ещё не сеньор и растете вы сами часто не вырастите опасность дикая, когда вы считаете себя гением среди тех, кто не знает. Очень многие через это проходят очень многие на этом спотыкаются. Дальше. Не провоцируйте частности проекта на общее. В любом частном проекте есть костыли есть свои ошибки есть свои странные решения в любом на любом уровне команды это бывает всегда не значит, что все проекты так делаются. Дальше. Если всегда так делали просто забудьте про это. Game Dev развивается как и любые подходы нету устаканившихся вещей нет серебряной пули не одно решение в движке не одно решение архитектурной не является серебряной пулей не ECS, не геймплей теги никакие абилки экспериментируйте и не доверяйте никому даже мне И уж тем более как бы любым технорям поэтому на этой ноте Всем спасибо и я буду рад ответить на вопросы.