

Формирование команды технарей

Harj Taggar & Ammon Bartram, TripleByte

Перевод: проф переводчик

Корректировка: опечатки. Расшифровка сгенерирована по титрам

Редактирование: Курлянова Дарья

 [Читать тезисы](#)

 [Читать конспект](#)

 [Смотреть видео с русскими титрами](#)

 [Оригинал видео](#)

 [Просмотреть исходники титров](#)

Это два абзаца статьи: ищем волонтеров и экспертов-волонтеров

1. Переведено волонтерами [Y Combinator по-русски](#), инициированного школой стартапов RUSSOL. Цель инициативы - передать русскоязычному пространству опыт создания и развития компаний вроде airBnB, MixPanel или Twitch. Волонтеры также проводят бесплатные просмотры переведенных лекций в инкубаторах, коворкингах, и университетах Пензы, Саратова, Кишинева, Астаны и др.

[Участвовать в просвещении, получить знания и перенять опыт](#)

2. Общественная [онлайн-школа стартапов RUSSOL](#) дважды в год проводит публичные бесплатные лекции, где на примере основателей компаний YouDo, Kabanchik, Флакон, LETA, Cinemood узнаете, как находить идеи, первых клиентов, создавать продукт, привлекать инвестиции. Школа ищет экспертов, спикеров и партнеров на 2019-2020 год, чтобы делиться своим опытом со слушателями.

Участие на добровольных началах. [Передать свой опыт другим](#)

Джефф: Пока загружаются слайды... Нет темы, которая должна больше занимать ваши мысли, когда вы строите компанию, чем подбор команды, которая сделает вашу компанию успешной на пути развития. Хардж и Аммон из Triplebyte, выпускники YC, расскажут о построении в наши дни и в нашу эпоху, наверное, ключевой части этой команды - команды разработчиков. Итак, поприветствуем... Хардж, ты начинаешь, да? Харджа!

Хардж: Отлично, спасибо всем за прием! Я Хардж. Я один из сооснователей Triplebyte вместе с Аммоном. До этого я был партнером Y-Combinator. На запуск Triplebyte меня в том числе вдохновило то, что, как я заметил, после выпуска из Y-Combinator и привлечения первых инвестиций, у всех вставала проблема номер один - найм персонала и особенно - разработчиков, потому что это один из самых непростых вызовов в отношении найма.

Поэтому, работая над Triplebyte, который является маркетплейсом для найма и используется разработчиками для поиска новых мест работы, Аммон и я собрали много данных о том, что хорошо работает, когда речь идет о найме разработчиков. Лично я фокусировался больше всего на том, чтобы проводить время с компаниями, помогая им продумывать стратегию нахождения разработчиков и, конечно, эффективно использовать Triplebyte. А Аммон провел много времени, продумывая в деталях, как оценить разработчика. Как оценить его инженерные навыки и ответить на вопрос, "хороший ли это разработчик или нет"? Поэтому мы собираемся поделиться этим и применить здесь стратегию "разделяй и властвуй".

Мы поговорим про четыре основных темы: Где искать разработчиков? Когда следует начинать думать об использовании рекрутеров? Это то, с чего я планирую начать. Затем мы поговорим об оценке технических навыков, об этом расскажет Аммон. И затем я закончу рассказом о процессе выставления предложений и заключении сделок, который ведет к тому, что люди действительно присоединяются к вашей команде.

Но прежде чем мы начнем что-либо из этого, я хочу честно предупредить и убедиться в том, что вы готовы к тому факту, что найм - это настоящий и конкретный отстой. Это необычайно трудный и болезненный процесс в силу многих причин, о которых я детально расскажу.

Во-первых, он занимает много времени. Большое время потребуется просто для того, чтобы убедить кого-то хорошего всего лишь поговорить с вами. А у основателя, как вы знаете, времени всегда не хватает. Баги, которые надо пофиксить, продажи, клиенты, с которыми надо контрактироваться, различные происходящие процессы. И найм никогда не кажется самой срочной задачей высшего приоритета, и очень легко протормозить и отложить его на потом. Но если вы будете это делать долго, вы не сможете масштабироваться, и ваш стартап не будет расти. Придет кто-нибудь еще, сделает это и займет рынок.

Во-вторых, найм требует выполнения повторяющихся процессов. Пока Тайлер показывал свою презентацию, я за кулисами поговорил с Джеффом о моей, и о том, продажи и найм имеют много общего. На самом деле, в этом очень похожи как продажи и найм, так и фандрайзинг, и многие другие вещи, которыми вы занимаетесь, как основатель стартапа. Большая часть этого - на самом деле непрекращающиеся продажи. А продажи, как и говорил Тайлер, включают в себя много повторяющихся задач. Поэтому найм потребует много переписки, большого количества встреч за кофе,

множества звонков и интервью. И большая часть из этого закончится тупиком и станет явной потерей времени. Но вам надо продолжать это делать. И наступит момент, который разобьет ваше сердце. Вы непременно столкнетесь с отказом людей, которых вы очень хотели нанять, тех, кто идеально подошел бы вам для помощи в достижении целей роста, но, оказывается, они серьезно не собирались оставить свою комфортную работу в крупной компании ради вашего удивительного, но рискованного стартапа.

Итак, будьте готовы ко всему этому, и, когда вы продумываете построение своего процесса найма, я советую вам думать о нем как о воронке (продаж), которую вы создаете и которая состоит из трех частей. Верхняя часть воронки - это получение, что означает нахождение людей, которые могли бы неплохо вам подойти. Вторая часть - это отбирающий просмотр, то есть ответ на вопрос: "Хотите ли вы нанять этого человека". И последняя часть воронки - это закрытие сделки, выставление предложения и подтверждение его.

Я начну с обсуждения некоторых стратегий для построения верхней части вашей воронки найма. Я советую вам смотреть на 5 мест, в которых вам стоит искать, когда вы нанимаете инженеров. Я проговорю "за" и "против" каждого из них. Это - личные сети, маркетплейсы для найма, LinkedIn/ GitHub, сайты с объявлениями о работе и встречи по интересам (митапы). Я расскажу о том, как вам получить максимум от каждого из этих мест. И этот список ранжирован или отсортирован в порядке того, с чего, по моему мнению, вам стоит начать и где фокусировать свое внимание и энергию, и до тех мест, где стоит фокусироваться меньше всего.

Итак, начнем с личных сетей. По моему мнению, личные сети - это лучшее место, откуда стоит нанимать, особенно в ранний период, когда вы наняли всего несколько человек. Причина в том, что каждый раз, когда вы решаете нанять кого-то, вы на самом деле задаете себе два вопроса. Первый: "Есть ли у человека навыки, которые мне от них нужны, чтобы выполнять работу?" И второй: "Можете ли лично вы эффективно работать с этим человеком."

Когда вы - большая компания, вы можете сфокусироваться на первом вопросе, так как в большой компании так много людей и команд, что где-то какая-то команда сможет эффективно работать с любым кандидатом. Но когда вы малы, это не так. То, сможете ли вы с кем-то работать или нет, - это большой фактор успеха. И если вы на ранней стадии нанимаете неправильного человека, это может быть буквально смертельным для стартапа. Поэтому когда вы нанимаете того, с кем раньше работали, либо нанимаете того, кто работал с человеком, которому мы доверяем, вы снимаете риски того, что вы не сможете работать вместе что очень важно учесть на ранних стадиях.

Это звучит как очевидный совет, и однако я удивлен, как часто основатели все еще не используют эффективно свои личные сети, когда нанимают людей. И я думаю, что этому есть две причины. Во-первых, они не используют какой-то определенный процесс для исчерпывающего поиска среди людей, которых они потенциально могут нанять. А во-вторых, они на самом деле не задают тот самый главный вопрос. Что

обычно имеет причиной страх того, что друзья тебя отвергнут. Действительно, быть отверженным абсолютным незнакомцем легче, чем спросить своего лучшего друга и услышать: "Ну, я не уверен, что это хорошая идея". Кроме того, вы можете беспокоиться о том, что случится с дружбой, если стартап не взлетит.

Похоже, что разговор с кем-то лично знакомым, запускает больше сомнений, чем разговор с тем, кого вы не знаете. Но правда состоит в том, что вам надо забить и сделать это. Если вы хотите успеха своему стартапу, поиск людей из тех, которых вы знаете - это страшно важный ресурс, и вы просто должны задать им вопрос. Советую вам здесь придерживаться конкретного процесса. Начните с того, что просто составьте список всех хороших разработчиков, которых вы знаете, неважно, доступны они или нет. Это не имеет отношения к делу. Пускай они только что продали свою компанию за миллиард долларов, все равно включайте их в список. Затем свяжитесь буквально с каждым из этого списка, чтобы встретиться и непременно спросите их, присоединяться ли они к вам. Насколько бы это вам ни казалось сумасшедшим или нереальным, обязательно задайте этот вопрос. Если они скажут "нет" или будут сомневаться, снизьте риск этой ситуации, сказав: "Не хочешь просто зайти к нам в офис и посмотреть, над чем мы работаем?" Если офис - это ваша квартира, это абсолютно нормально.

Но продолжайте предлагать, пока вы хотя бы покажете им что-то, что вы сделали. Поработайте над тем, чтобы их убедить. Если это не сработает, если они скажут "нет", и это будет определенное "нет", то спросите их - если бы они были на вашем месте, кого бы они попробовали привлечь. Составьте список, и повторите с ними в точности тот же процесс. Этот процесс просто никогда не оканчивается.

Я знаю основателей публичных стартапов, собственно уже не стартапов, а публичных компаний, которые все еще делают это ежедневно. Это ключевая вещь, которую вам надо встроить в себя, как основателя стартапа. По мере того как ваша компания масштабируется и растет, и вы начинаете собирать команду, вы начинаете стучаться в личные сети вашей команды. И я рекомендую это делать через командные мероприятия с мозговым штурмом про потенциальных кандидатов. Они часто называются отборочными вечеринками.

Я советую организовывать их так: собираете всех вместе, рассылаете общую таблицу и описываете роль, на которую вы набираете человека. Если это роль разработчика, опишите в деталях, кого вы ищете, кто может быть примером кандидата, какие навыки и качества вам было бы прекрасно в нем увидеть? И затем попросите буквально каждого в течение 30-45 минут просмотреть свои контакты на LinkedIn и Facebook, или же где-то еще, прямо там и тогда подумав о каждом, кто мог бы подойти, и записав его в таблицу.

Когда это сделано, я у себя в Triplebyte потом лично связываюсь с каждым в списке, кажется ли он хорошим кандидатом или нет. Таким способом мы наняли несколько по-настоящему прекрасных людей. Этот способ работает очень, очень хорошо. И кроме того, вы можете сделать из этого повод повеселиться, не так ли? Мы делаем

такие встречи в конце недели, прямо перед пятничным общим собранием. Приносим туда еду и напитки, а еще вы можете предложить бонусы своей команде за рекомендации, чтобы мотивировать их на это. Только обязательно используйте [неразборчиво], придерживайтесь исчерпывающего процесса, задавайте вопрос людям, которых вы знаете, и затем, когда масштабируетесь, обращайтесь к личным сетям вашей команды.

После того как вы исчерпаете вашу личную сеть в поисках лидов, следующее место, где бы я начал искать, это маркетплейсы для найма.

Маркетплейсы для найма - сравнительно новое изобретение. Они стали популярны в течение нескольких последних лет, так как стало труднее нанимать инженеров, используя традиционные методы, такие как контакты на LinkedIn или GitHub. И я собираюсь больше поговорить об этом позже. Я думаю о маркетплейсах, что они работают очень похоже на сайты знакомств. Идея в том, что есть инженеры, создающие профили, компании, создающие профили, и те и другие рекламируют себя с лучшей стороны. Вы отправляете друг другу сообщения, и решаете, стоит ли встретиться лично, и, если все складывается, нанимаете человека. Однако динамика такого маркетплейса такова, что спрос на хорошие инженерные таланты далеко превосходит предложение. Поэтому обычно именно компании гораздо активнее в контакте с кандидатами. Кандидаты получают множество запросов, и именно они - кандидаты, или разработчики, выбирают, с кем они хотят поговорить, а с кем нет.

Большим преимуществом маркетплейсов, особенно на ранних стадиях, является то, что они помогают нанимать очень быстро, потому что большинство кандидатов на маркетплейсах находятся в активном поиске места, куда пойти работать прямо сейчас. Можно очень быстро позвонить им и рассказать о себе. И если вы эффективно выполняете закрытие сделки, вы можете существенно снизить свое время, как основателя, на найм, Что очевидно, круто.

Однако, недостатком маркетплейсов является конкуренция. Инженеры получают предложения от нескольких компаний сразу. Поэтому вам придется очень эффективно убеждать их присоединиться, если вы хотите их получить. И второе - это может быть дорого. Большинство маркетплейсов работают по схеме оплаты за найм, которая может составлять 15-20% от зарплаты за первый год. Это дешевле, чем рекрутинговое агентство, но все же значительная сумма, если вы - стартап на ранней стадии.

Конечно, я здесь пристрастен, так как Triplebyte - это маркетплейс найма, но я бы сказал, что три основных места, которые всплывают в разговоре, по крайней мере, когда мы говорим с заказчиками, Triplebyte, Hired и Vetterly. Вам следует их попробовать. Все они позволяют стартовать бесплатно, так что добро пожаловать.

Про Triplebyte я бы отметил, что мы отличаемся в основном тем, что у нас более хорошие кандидаты. И мы измеряем это как процент кандидатов, с которыми компании проводят интервью через Triplebyte, и которым затем делают предложение. И этот показатель обычно в два раза выше, чем при поиске в других источниках. Как общее

замечание: найм - это воронка, и вы автоматизируете вашу воронку, поэтому вам следует уделять внимание проценту кандидатов, которые проходят каждый следующий шаг в вашей воронке.

На ранних стадиях у вас не так много кандидатов, поэтому вы не можете применять такой уж научный подход к этому. Тем не менее, начинайте собирать эти данные и встраивать это в ваши привычки по отношению к найму.

Третий источник, куда я советую пойти и посмотреть - это LinkedIn и GitHub. Фактически, это самые большие онлайн-каталоги инженеров в мире. Большая часть сотрудников крупных компаний нанимается через команды технических рекрутеров, которые выходят на разработчиков через LinkedIn или GitHub, находят тех, кто подходит по определенным ключевым словам, и посылают им "холодные" сообщения. И они рассчитывают на очень большие объемы.

Так, технический рекрутер вполне может послать больше 100 сообщений в день просто в надежде получить несколько откликов. Там имеет место такая динамика, что, особенно последние несколько лет, технических рекрутеров становится все больше, и все больше сообщений отправляется с таких платформ, поэтому доля ответивших падает для всех. Следовательно, особенно для стартапов на ранних стадиях, это будет требовать от вас много времени на отсылку многочисленных сообщений, чтобы получить немногих интересных кандидатов.

Поэтому чтобы это заработало для вас, по моему мнению, требуется не такой подход с большими объемами, который подходит для крупных компаний. А другой подход, при котором вы проводите время, по-настоящему исследуя, читая в деталях профили и страницы кандидатов в LinkedIn, GitHub, обращая внимание в деталях на работу, которую они выполняли. И отсылаете небольшое количество персонализированных, нацеленных сообщений подчеркивая в сообщениях, почему вы считаете, что человек подойдет именно вашей компании, и позволяя кандидату увидеть, что вы прочитали его профиль, и вы интересуетесь им, как личностью, а не рассылаете спам. Однако это не значит, что сообщение должно быть очень длинным. Я советую вам делать сообщение кратким и лаконичным. Ключевым является то, что показывает, что вы прочитали его профиль.

В заключение замечу, что стоит посылать email, а не мгновенные сообщения. Если вы подпишетесь на LinkedIn Recruiter Lite, что обойдется вам около \$120 в мес, то сможете иметь доступ к Connectifier, это плагин к Chrome, в котором очень просто вытаскивать адреса email из любых профилей LinkedIn. И вы точно увидите гораздо более высокий процент ответов по email, чем через сообщения. Поэтому точно, определенно используйте email.

Четвертое место, в котором я бы искал разработчиков - это сайты с объявлениями или предложениями о работе. Для стартапов два основных - это Stack Overflow Jobs и AngelList. Я не включил сюда предложения на HackerNews Jobs, потому что они доступны только для компаний YC. Однако предложения на HackerNews уникальны

тем, что там присутствуют разработчики особенно высокого класса, и я бы поставил этот сайт на второе место в списке, если бы он был самостоятельным источником для поиска инженеров.

Вообще, доски объявлений страдают от преобладания количества над качеством. А их преимущество в том, что вам не нужно проводить огромное время, публикуя там. Недостаток однако в том, что потери времени поджидают вас позже, потому что большинство кандидатов, которых вы получите, будут иметь абсолютное недостаточные компетенции. Вы получите много предложений от людей, которые вообще не являются разработчиками, и у вас займет много времени прочитать все эти резюме и предложения, чтобы найти одного или двух хороших кандидатов.

Поэтому, чтобы максимизировать ваш результат и число хороших кандидатов, я советую сосредоточиться на том, чтобы сделать ваши описания вакансий уникальными и интересными. И имейте в виду, что большинство описаний вакансий в интернете написаны кем-то из отделов найма персонала или маркетинга и используют бюрократический стиль, который особенно не привлекает аудиторию разработчиков, не так ли? Поэтому, как основатель стартапа, вы можете экспериментировать и привносить в описание вакансий живое и личное.

Один из способов, который вы можете попробовать - это писать от первого лица, говоря про вашу личную историю, почему вы создали компанию, почему вы заряжены этой миссией, и пусть это будет как... Донесите эмоции и страсть.

Другой способ для вас - подумайте, в чем уникальность культуры вашего стартапа. В чем особенные вызовы в технологиях или продукте, с которыми вы сталкиваетесь? Если вы добавите конкретики об этом, это тоже выделит вас, так как большие компании склонны говорить в общих чертах и туманно о том, что ожидает работающих у них людей.

И последний источник, о котором я расскажу, это настоящие, физические встречи по интересам (митапы). Они не кажутся мне очень эффективными, и это скорее длительное вложение. Их результаты не очень-то высокие. Очные встречи собирают не столь уж много участников, и, по правде говоря, люди приходят туда больше за бесплатной едой и напитками, чем за поиском места работы. Поэтому маловероятно, что вы найдете там реально знающего кандидата, который при этом активно ищет работу, и к тому же увлечется вашим стартапом. И вам лично придется быть очень эффективным в разговорах с незнакомыми людьми и убедительным для них, чтобы это могло сработать.

Я включил этот источник в список по той причине, что я действительно знаю основателей стартапов, которые добились успеха через встречи по интересам. Однако их мало и они далеко. Если вы будете пробовать этот способ, я советую сосредоточиться на технических встречах по интересам. То есть у местной группы Closure programming где люди с ноутбуками собираются вместе для работы на

проблемами, больше шансов стать источником разработчиков, чем у конференции Dreamforce.

В заключение добавлю, что вы можете устроить группу по интересам в собственном офисе. Соответственно, вы можете совместить этот способ с привлечением через личные сети и использовать такие встречи как повод пригласить друзей зайти в офис или пригласить их собственных друзей-разработчиков зайти в ваш офис.

В Triplebyte один из разработчиков - фанатичный пользователь eMax, и он проводит митапы eMax Bay Area в нашем офисе. Это похоже на... Это пока не сработало для нашего найма, но является хорошим способом просто встретиться для построения сети хороших разработчиков, которая может стать полезной в будущем. Поэтому такой способ точно стоит иметь в виду.

Теперь я собираюсь поговорить о том, когда вам следует думать об использовании рекрутеров. По правде говоря, нет никакого жесткого правила по поводу того, когда вам стоит нанять технического рекрутера. Я видел компании из меньше чем десятка человек, которые нанимали рекрутера. Также я видел и компании, которые ждали, пока у них будет 50+ человек.

Я выскажу свое мнение, когда это надо делать, скорее как практический прием. Во-первых, я думаю, что вы точно должны дождаться того, когда вы наняли хотя бы одного разработчика, прежде чем думать о найме рекрутера, независимо от размера компании. Так как здесь применим общий совет всем стартапам. о том, что, когда вы нанимаете, вам будет полезно некоторое время делать эту работу самому, чтобы вы почувствовали боль и поняли, о чем это. А также чтобы вы поняли в деталях, что делает человека подходящим для этой роли конкретно в вашем стартапе, перед тем как вы наймете рекрутера. Вы сможете лучше оценить его, и, если вы почувствовали боль самостоятельного рекрутинга, то сможете лучше нанять правильного рекрутера для вас.

А вторая причина, я бы сказал, такая же как в отношении продаж. Для вас очень полезно выйти и попытаться рассказать и убедить людей присоединиться к вашему стартапу, чтобы вы просто сами поняли, какой посыл работает, а какой нет. Потому что, как основатель стартапа, вы всегда продаете и никогда не знаете, когда наткнетесь на кого-то, кто может стать по-настоящему прекрасным приобретением. А если вы уже практиковались в убедительном рассказе для разработчиков, то ваша речь будет всегда наготове. Поэтому я советую всегда сделать первый найм самому перед тем, как делегировать это техническому рекрутеру.

Во-вторых, я бы ориентировался на хороший темп найма персонала. Что-то типа найма одного разработчика в месяц в течение следующего полугода, перед тем как привлекать рекрутера. Иначе ему скоро станет нечем заняться. И наконец, как практическое правило, тратите ли вы больше чем 50% своего времени на поиск персонала, включая все то, что я говорил раньше, а также первые звонки, просмотры и приглашения на личные встречи? Более 50% - это примерно та граница по времени,

когда вам следует начать думать о привлечении помощи в этом. Потому что 50% - это примерно то время, которое имеет смысл тратить на найм.

Сами рекрутеры бывают примерно трех типов. Рекрутеры по контракту с почасовой оплатой, которые могут делать все - от холодных сообщений на LinkedIn до первичных просмотров по телефону. Далее, бывают внутренние рекрутеры, то есть вы нанимаете технического рекрутера на полный рабочий день, и он работает как член вашей команды. И наконец, бывают агентства. Агентства - это по сути команды продающих людей, которые связываются с большим количеством разработчиков на LinkedIn или где-то еще, а затем продают их резюме как можно большему количеству компаний. Они обычно просят 25-30% от зарплаты первого года, если вы нанимаете разработчика по резюме, которое они вам прислали. Они часто стараются быть очень полезны для компании, и поэтому очень вовлечены в то, чтобы дать вам информацию, которая поможет вам понять разработчика, однако они рассылают эту же информацию нескольким компаниям одновременно.

Мой совет в этой части - когда вы подходите к точке, когда чувствуете необходимость в помощи или рекрутере, начните с рекрутера по контракту и сфокусируйте его на поиске кандидатов. Пусть он сфокусируется на контакте с разработчиками на LinkedIn и GitHub, а его основным результатом для вас пусть будет заполнение вашего календаря созвонами с подходящими кандидатами. А вы будете рассказывать и убеждать.

Когда вы дойдете до точки, где объем такой работы станет для вас слишком большим, я бы предлагал нанять внутреннего рекрутера на полный рабочий день и подготовить его для выполнения первоначальных продающих звонков. Теперь они будут и получать контакты, и выполнять первые звонки, а также организовывать очные интервью для вас.

Итак, давайте резюмируем. Мой план найма для стартапа, если я собираюсь построить команду разработчиков, будет таким: начните с того, чтобы исчерпать вашу личную сеть, проведите много времени, приглашая людей пообедать или выпить кофе и - предлагая им присоединиться.

Два - экспериментируйте с маркетплейсами найма. Там полезность будет зависеть от того, насколько эффективно вы доносите информацию о своей компании. Но даже если вы не наймете там никого, вы получите значимый опыт продажи реальным разработчикам и кандидатам, а также научитесь тому, что именно в вашей компании вызывает резонанс этой аудитории.

Третье - проведите некоторое время в конкретных, персонализированных целенаправленных коммуникациях с разработчиками на LinkedIn и GitHub, делая ваши сообщения действительно персональными.

И наконец, рассматривайте сайты вакансий, встречи по интересам и очные встречи как фоновый процесс, который вы ведете, не ожидая оттуда множества нанятых сотрудников, но создавая значимый канал для будущего.

Супер. Мы закончили первую часть. А сейчас Аммон расскажет о том, как просматривать и оценивать разработчиков технически, после чего я вернусь и завершу нашу тему рассказом о выставлении предложений и закрытии сделки.

Аммон: Отлично! Спасибо, Хардж. Итак, я Аммон, вместе с Харджем я являюсь сооснователем Triplebyte, а до этого я запустил Socialcam вместе с Майклом Зибелем, а также был одним из первых сотрудников Twitch. Я собираюсь рассказать о шаге отбирающего просмотра.

Итак, как же определить опытных разработчиков для вашей компании. Первый вопрос - почему вам стоит вообще верить мне в этих вопросах. Первый ответ: я провел очень много интервью. С основания Triplebyte я интервьюировал лично более 1000 разработчиков. Но я думаю, что лучшим ответом является то, в чем Triplebyte имеет особое преимущество. Мы видим, как кандидаты проходят интервью во многих различных компаниях. Мы видим, как один и тот же кандидат ведет себя в разных компаниях, и это дает нам множество данных, которых, я полагаю, нет ни у кого другого. Именно из этих данных родились те советы, которые я буду давать сегодня.

Перед тем, как погрузиться в эти вопросы, я хотел бы пройти по базовому процессу найма, который используют большинство компаний. На самом деле, он довольно стандартный. 90% компаний используют эти базовые шаги для просмотра кандидатов. Первый шаг - просмотр резюме. Кто-то подает в компанию заявку, присылает резюме, рекрутер просматривает это резюме и решает, является ли кандидат кем-то в основном подходящим.

Затем звонок рекрутера. Обычно это 30-минутный телефонный разговор рекрутера, который спрашивает об опыте кандидата, оценивает культурную совместимость и проверяет, действительно ли кандидат заинтересован в компании. Потом происходит технический отбор по телефону. Он проходит от 30 минут до часа, в течение которых разработчик обычно решает конкретную программную задачу. Это похоже на FizzBuzz-задачи, или же немного сложнее. Это зачастую делается на каком-нибудь синхронизируемом TextPad.

Затем, но не обязательно, разработчик берет домашний проект. То есть какой-нибудь существенный проект, который кандидат выполняет самостоятельно и отправляет результат в компанию на оценку.

И, наконец, происходит интервью на месте. Это означает, что кандидат приходит в офис и участвует в серии из от 3 до 6 часовых сессий с разработчиками компании, которые обсуждают отдельные задачи. И затем собирается встреча по решению. Обычно, на следующий день после того, как кандидат вернулся домой, все, кто интервьюировал кандидата, а также специалист по найму, собираются в одной

комнате и рассказывают о своих впечатлениях, и затем принимают групповое решение о найме или отказе.

Вот некоторая статистика по этому поводу. Компании делают предложение от 2 до 8% всех разработчиков, подавших заявки. Но интересно, что, этап, на котором происходит такой отсев, различается от компании к компании. Например, есть компании, где 75% подавших заявку отсеиваются при просмотре на первом шаге, при проверке на культурную совместимость. Также есть компании, где почти все кандидаты доходят до финального интервью, на котором и происходит отсев. Также известно, что 95% нанимаемых людей остаются работать, что означает, что 5% нанятого технического персонала увольняются в ближайшие месяцы.

С точки зрения кандидата то, что мы видим представляет собой распределение вероятности успеха на интервью. Так, несколько процентов программистов, превосходящих других по уровню опыта, получают предложение о работе после большинства интервью. Но есть куча программистов где-то в середине, которые проходят интервью и получают предложения после примерно 15-20% пройденных интервью.

Интересно, что нет того, кто бы всегда проходил интервью успешно. Не существует волшебных разработчиков, которые получают предложение после любого интервью, которое проходят. И это приводит нас к тому, что, как я думаю, является наибольшим вызовом для разработки процесса интервью: это не целостность. Это "шум" для процесса интервью. Идея в том, чтобы понять, является ли ваше интервью фундаментально повторяемым и значимым.

Если бы вы могли как-нибудь повторно интервьюировать своих коллег, сотрудников, каждого кто проходил ваше интервью за последний год, если бы вы могли повторно интервьюировать их людей, сколько из них прошло бы снова? Это пугающий вопрос, но интересно, что мы смогли собрать некоторые данные об этом. Я вычислил статический показатель, называемый "надежность рейтинговых оценок" между вообще всеми интервьюерами всех компаний на Triplebyte. Это означает, что данный показатель - статистическое измерение того, насколько разные интервьюеры склонны соглашаться в том, какие кандидаты лучшие. Этот показатель принимает значения от 0 до 1, где 0 означает отсутствие согласия, или же ту величину согласия, которую можно ожидать увидеть в случайных данных, а 1 означает полное согласие. И я обнаружил, что согласие измеряется величиной чуть большей чем 0.

На первый взгляд, это очевидно гораздо ближе к отсутствию согласия, чем к полному согласию. Но чтобы создать некоторый контекст, я вычислил тот же показатель для набора данных об онлайн-обзорах фильмов. И я получил уровень согласия почти такой же, фактически всего лишь чуть выше. В итоге выходит, что интервьюеры в оценке того, какие кандидаты лучшие, согласны примерно в той же степени, что и пользователи Netflix согласны по поводу того, какие из фильмов лучшие. Это пугает.

Результаты интервью фундаментально "зашумлены", и в большей степени... данные показывают их более зашумленными, чем считают большинство специалистов по найму. Отсюда первый вопрос: зачем вообще делать интервью? Если интервью до такой степени "зашумлены", зачем их проводить? Почему бы нам не использовать что-то типа испытательных сроков для просмотра разработчиков? Да, я считаю это действительно прекрасной идеей. Это почти точно дает гораздо больше. Если ты можешь поработать с кем-то неделю, ты почти точно сможешь получить гораздо лучшее представление о том, является ли он хорошим сотрудником, чем то, которое можно получить в процессе трехчасового интервью с ним.

Проблема в том, что разработчики по большей части не хотят испытательных сроков. Мы провели в Triplebyte исследование, и оно показало, что только 20% разработчиков хотят работать с испытательным сроком. И там на самом деле есть обратная выборка. Большинство лучших программистов находятся внутри тех 80%, которые предпочитают проходить стандартное техническое интервью, так как оно отнимает гораздо меньше времени и в целом гораздо быстрее. Поэтому я думаю, что найм через испытательный срок - прекрасный вариант, и вы можете его предлагать как вариант, но, если вы не хотите распугать большинство лучших программистов, вам придется использовать стандартный, традиционный процесс интервью.

Сегодня я хочу поговорить о конкретных советах, которые вы можете использовать, чтобы снизить "зашумленность" в традиционных интервью. И я собираюсь пройти 7 основных пунктов.

Пункт 1, первый способ снизить "шум" на интервью - это решить, какие навыки важны для вашей компании. Программист может иметь множество совершенно различных компетенций. Например, кто-то очень производителен, кто-то работает медленно и внимательно, пишет прекрасные тесты и не допускает багов. Еще кто-нибудь может быть силен в математике и информатике, либо же иметь глубокие знания о ядре Linux или о настройке расписаний для операционных систем реального времени, или о чем-то еще. Если вы как основатель не решите, какие из этих навыков имеют значение для вашей компании, то проводящие интервью решат за вас. Они будут использовать свои стандартные вопросы на интервью и будут отказывать людям, которые плохо отвечают на них, даже если это вопросы в областях, которые не нужны для вашей компании.

Это на самом деле источник большей части шума в интервью. Каждый разработчик предрасположен считать те вещи, которые он знает лучше всего, самыми важными, которые все должны знать. В отсутствии конкретных указаний сверху о том, на что смотреть, он будет отказывать людям, исходя из областей, неважных для компании. Поэтому мой первый совет - последовательно задать такие вопросы себе самим, перед тем как начинать найм.

Мы видим большинство несогласий между интервьюерами в следующих осях. Во-первых, нужны ли нам программисты, работающие быстрыми итерациями, либо же кто-то, кто делает внимательно и вдумчиво? Хотим ли мы нанять тех, кто мотивируется

решением трудных технических задач, либо созданием красивого продукта для пользователей? Важно ли, чтобы пришел кто-то с навыками в конкретной технологии, например, в конкретном языке программирования? Либо вы хотите нанять умного человека и позволить ему изучить ваш стек технологий в процессе работы? Важными ли будут образование в области информатики и умение работать с алгоритмами, либо такие вещи не относятся к делу? И наконец, какая еще конкретная экспертиза требуется от нанимаемого человека?

Совершенно нормально на некоторые из этих вопросов ответить двояко. Не обязательно задавать единственный профиль нанимаемого сотрудника. Важно лишь понять, что имеет значение, даже в случае нескольких профилей. Это позволяет настроить весь остальной процесс найма и убедиться в том, что вы не откажете людям за провалы в ненужных вам областях.

Пункт 2. Второй способ снижения "шума" - это структурированные интервью. Чтобы дать определение, неструктурированное интервью в свободной форме - это интервью, где проводящий интервью заходит в одну комнату с кандидатом и затем задает вопросы, следуя своей интуиции. В зависимости от ответов кандидата, он задает последующие вопросы и пытается понять, как воспринимать кандидата, и подходит ли он компании. В итоге он принимает решение "да/нет", "нанять/отказать" на основании интервью. Это составляет контраст со структурированным интервью, при котором проводящий интервью приходит с вопросом, который он собирается задать и определенными критериями, по которым он будет оценивать.

Интересно, что все считают, что кандидаты воспринимают свободную форму лучше. Практически все проводящие интервью предпочитают свободную форму и полагают, что такая форма дает более точные результаты. И многие из кандидатов, если их спросить, действительно скажут, что предпочитают, чтобы их интервьюировали свободным способом. Однако интересно то, что все данные говорят об обратном. Структурированные интервью просто дают более надежные результаты. Они лучше предсказывают успех на работе, и нет никакой причины к тому, чтобы их не использовать.

Нам всем следует использовать структурированные интервью. Сейчас я пройдуся по некоторым примерам того, как это может происходить. Во-первых, вам следует сохранять процесс постоянным. Так как цель интервью в основном в том, чтобы оценить различие между кандидатами, и, если вы не сохраняете весь процесс постоянным от кандидата к кандидату, вы вводите "шум". Таким образом, нет никакой причины не задавать каждому кандидату на одну роль в точности одного и того же набора вопросов. Я полагаю, что причиной непопулярности такого способа является то, что сами проводящие интервью считают это скучным. Все, что я могу сказать, - смиритесь с этим. Вы должны это сделать.

Второе, что важно для структурированного интервью: вы должны дать проводящим интервью определенные критерии для оценки. Вместо того, чтобы поместить их в комнату и сказать: "Решайте, подойдет ли этот человек нашей компании", скажите:

"Нас интересует результативность и знания о бэк-энд системах, и поэтому ваша цель - задать на интервью этот конкретный вопрос и оценить кандидата по результативности кодирования и уровню знаний бэк-энд систем".

На самом деле есть прекрасное исследование по этому вопросу. В нем делается вывод о том, что худшие предубеждения, возникающие на интервью, еще более провоцируются тогда, когда проводящий интервью пытается принять глобальное решение. То есть, если кто-то должен принять глобальное решение: "Является ли этот человек хорошим сотрудником?", вступают в игру такие факторы, как "похож ли он на кого-то, кого я знал", или раса, или пол кандидата. Проводящие интервью гораздо проще игнорируют эти атрибуты тогда, когда им даны для оценки определенные критерии.

И, наконец, полезно унифицировать принятие решений. Чаще это относится к более крупным компаниям, но вам следует обеспечить то, что он один и тот же человек или одна группа людей принимает все финальные решения. Считайте интервьюирующих не принимающими решения, смотрите на них как на делающих заметки, ставящих оценки по критериям, после чего все эти заметки и критерии отправляются центральному человеку или центральной группе людей, которые принимают финальное решение. Идея здесь в консистентности, так как гораздо проще быть консистентным, если один человек принимает все решения.

Пункт 3 для снижения "шума" - это использовать более качественные вопросы на интервью. У меня есть некоторые подсказки по этому поводу. Во-первых, стоит избегать вопросов, которые требуют от кандидата новых инсайтов. Вместо них пусть будут вопросы, создающие ступеньки для того, чтобы последовательно прийти к решению. Это общая идея, а в качестве практического правила, которое мне нравится, стоит спросить себя, может ли ответ на этот вопрос быть заранее подсказан? То есть, если это вопрос про отдельный факт в решении, который кандидат мог знать заранее или который ему мог подсказать приятель, либо который он мог просто прочесть на Glassdoor, то, скорее всего, это плохой вопрос.

В качестве примера приведу классический вопрос. "Представьте себе, что вы стоите внизу лестничного пролета, и каждым шагом можете либо подняться на следующую ступеньку, либо перепрыгнуть через одну или две ступеньки. Сколько существует уникальных путей до верхней ступеньки лестничного пролета?" В итоге оказывается, что решением этой задачи является, как ни странно, последовательность Фибоначчи. Если кто-то уже знает об этом, то, конечно, сразу ответит на такой вопрос. А если не знает, то, вероятно, затруднится ответить, задумается и провалится в "кроличью нору" бесконечных размышлений. Это пример плохого вопроса, для решения которого требуется новый инсайт.

Пример хорошего вопроса: "Пожалуйста, реализуйте мне игру "Четыре в ряд". В этом случае существует последовательность шагов, каждый из которых довольно прозрачен, которая ведет к решению, и при этом нет ничего, что приятель мог бы

рассказать кандидату за 10 минут, и что дало бы серьезное нечестное преимущество в игре "Четыре в ряд."

Поэтому еще одна идея, связанная с этой, состоит в том, что лучше использовать вопросы, состоящие из нескольких шагов. Они не только приводят к задачам без новых инсайтов, но также помогают в том не редком случае, когда кандидаты, даже хорошие, "зависают" на интервью. Если ваша задача состоит из нескольких шагов, вы можете дать ему подсказку в одной части, и однако оставить достаточно, чтобы кандидат продолжил решать, собрался и продемонстрировал свой опыт. Если же ваша задача - "все в одном" и похожа на монолитный камень, то в случае, когда кандидат не может ее решить, и вам придется в этом ему помочь, получится, что он по сути провалил данный раздел вопросов.

Следует также избегать специализированных знаний. Скажем, если ваша цель - оценить общие навыки программирования, то следует задавать вопросы, которые используют такие вещи, как списки, хэш-таблицы или строки, а не вопросы, которые упоминают префиксные деревья. Если только, конечно, вам не требуется именно знание алгоритмических структур данных, и ваша цель состоит в том, чтобы каждый знал префиксные деревья. Если так, то спрашивать об этом нормально. Но если вы измеряете что-то другое, а частью задачи является префиксное дерево, то часть кандидатов будет с ним знакома, а другие нет, и это приведет к "шуму". Поэтому вообще говоря я полагаю, что лучше придерживаться классических, базовых концепций информатики.

Еще одно правило. Стоит отвести примерно в три раза больше времени чем требуется вам лично на решение задачи для кандидата. То есть, если вы выбрали вопрос, и сами решили его за 10 минут, то это вероятно хороший вопрос для интервью на 30 минут. Причина в том, что быть интервьюирующим гораздо проще, чем быть кандидатом. Гораздо проще задавать вопросы, и мы склонны недооценивать их настоящую сложность.

Мы провели реальное исследование этого вопроса. Мы прошлись по всем вопросам, которые мы задавали в Triplebyte, и мы смотрели, какие из них в большей степени коррелировали с успехом кандидата. И выяснили, что наиболее показательными были гораздо более простые вопросы, чем те, которые нам интуитивно кажутся показательными. То есть большинство проводящих интервью думают, что оптимальным является существенно более трудный вопрос, чем тот, который оптимален на самом деле.

Старайтесь задать не менее 4 вопросов на интервью. Идея в том, что каждый отдельный вопрос несёт в себе некоторое количества "шума". Возможно, человек уже сталкивался с этим вопросом? Или ему просто повезло правильно ответить? Если вы задаете больше вопросов, то на выходе получите более надежный результат.

И наконец еще одна подсказка. Нам в Triplebyte очень нравится такой вид вопросов, при котором мы даем кандидату задачу, а затем вместо того, чтобы ожидать поиска

решения, мы сообщаем ему решение задачи, и ставим ему целью взять данную идею и реализовать ее в коде. Таким образом, мы даем ему алгоритм и смотрим, может ли он реализовать его, а не требуем такой алгоритм разработать.

Четвертый путь снижения "шума" - игнорировать регалии кандидатов во время интервью. Под регалиями я понимаю такие вещи, как учился ли кандидат в знаменитом ВУЗе, или работал ли в известной компании. Я не утверждаю, что регалии не имеют значения. Они важны, не так ли? Они всегда показательны. Группа людей, раньше работавших в Google, действительно являются лучшими разработчиками, чем группа людей, которые там никогда не работали. Поэтому абсолютно законно принять этот факт во внимание, принимая решение о найме. Однако это не имеет отношения к его реальным навыкам программирования. И мой совет состоит в том, чтобы бы вы не были предвзяты из-за того, что кто-то пришел из Google, когда вы оцениваете его конкретные навыки программирования.

Поэтому я советую скрывать регалии кандидатов от проводящих интервью. Так как мы нашли, что это действительно сильно влияет на мнение интервьюеров. Если они знают, что кто-то имеет регалии, которые для них очень ценны, они с большей вероятностью интерпретируют результаты кодирования в лучшую сторону. "О, этот парень, ну, он не знал ответа на вопрос, но я уверен, что это лишь временный сбой. Поэтому скрывайте регалии от проводящих интервью. Давайте им возможность оценить умение программировать, а затем, вынося конечный вердикт на встрече по принятию решения, примите во внимание как регалии, так и навыки, показанные на интервью. Это поможет находить знающих программистов, у которых отсутствуют традиционные регалии. И тех людей, которые недооценены рынком. Умение находить таких людей - большое преимущество для вас как стартапа.

Пункт 5. Вам стоит подумать о количестве ошибочных негативов в результате интервью. Ошибочный негатив - это когда кто-то потерпел неудачу на вашем интервью, но при этом мог бы в итоге хорошо выполнять работу в компании. Противоположное ошибочный позитив - это когда кто-то успешно проходит ваше интервью, и вы его наняли, но он плохо проявляет себя на работе, и, вероятно, будет уволен.

Как ошибочный негатив, так и ошибочный позитив, стоят очень дорого. Если вы наняли не того человека, вам придется его уволить, это ужасно. Это вредит моральному духу команды, а также очень дорого просто в деньгах. Большая проблема. Однако, если вы - голодный стартап... Вы откатываетесь назад, если у вас не хватает разработчиков. Если вы отвергли человека, который мог бы присоединиться к вашей команде и быть продуктивным, это также очень дорого.

Я говорю здесь о довольно тонком вопросе, но я полагаю, что мы несколько предвзяты в том, что ошибочные позитивы... Мы очень хорошо понимаем их. Если мы найдем не того человека, мы будем испытывать боль в течение месяца или даже дольше, в лучшем случае. И мы в Triplebyte видели, как чаще всего интервьюеры слишком сильно верят в то, что люди, не прошедшие их процесс, являются плохими разработчиками, и не справятся с этой работой. Опыт не подтверждает этого.

Если мы посмотрим... Как я говорил ранее, ни один разработчик не может пройти успешно всех интервью. Значительная часть людей, которые провалили интервью, действительно устраиваются и эффективно работают в других компаниях. Поэтому я просто советую тем, кто разрабатывает процесс интервью, думать об ошибочном негативе и пытаться придать этому показателю некоторый вес в своих расчетах необходимого для найма уровня.

На самом деле, одна из моих целей в Triplebyte - достичь возможности настоящего измерения уровня ошибочного негатива. Потому что никто его не знает. Никто не знает, каков уровень ошибочного негатива на интервью, так как, чтобы его измерить, пришлось бы нанимать людей в случайном порядке и смотреть, как они справляются. Это очень дорого. Моя цель - достичь возможности измерять этот уровень.

Пункт 6 - это то, что лучше всего настраивать процесс на максимальный уровень навыков, который может принести кандидат, а не на средний или минимальный уровень. Когда кто-то приходит на интервью, он очень силен в одной области, а в других слабее. И наибольшее значение имеют самые сильные области. Любой может выглядеть глупо, не правда ли? Если вы зададите мне правильный вопрос, я точно буду выглядеть очень глупо, и это верно для каждого.

Проблема в том, что человек может пройти все интервью, и хорошо показать себя в некоторых полезных для компании навыках, но выглядеть глупо на одном из вопросов. И если конкретный интервьюирующий сделает блокирующее замечание о том, что кандидат глуп, это, возможно, введет в процесс "шум". С другой стороны, если кто-то показывает плохие навыки в области, важной для компании, безусловно, ему нужно отказать. Тем не менее, будьте открыты к тому, что каждый иногда выглядит глупо, и не отвергайте кого-то просто из-за глупого вида на одной из частей интервью.

Окей, последний пункт - это то, что вам стоит думать о том, что испытывает кандидат на интервью, когда вы готовите процесс интервью. Вам важно убедиться, что каждому кандидату, проходящему через ваш процесс, нравится ваша компания. И это так по нескольким причинам. Первая причина в том, что, если ему понравится процесс, то он с большей вероятностью примет предложение, которое вы сделаете. А это очень помогает на этапе закрытия сделки. Но что более интересно, это также делает ваш отбор более точным, потому что стресс оказывает серьезное влияние на результаты. Большой процент кандидатов испытывают серьезный стресс на интервью и не показывают своих лучших возможностей.

Вот несколько советов по уменьшению стресса. В том числе, позволяйте каждому приносить свой ноутбук, работать в своем окружении, писать на своем языке программирования, использовать свои инструменты. В этом случае они будут гораздо более продуктивны и будут находиться в состоянии гораздо меньшего стресса. Кроме того, учите проводящих интервью софт-навыкам: быть дружелюбными, предоставлять кандидатам перерывы, и, когда кандидаты не справляются с заданием, вмешиваться наименее стрессовым способом, не оскорбительным для кандидата.

Практическое правило по этому поводу, которое пришло еще от Old Joel из блога Software, состоит в том, чтобы каждый кандидат, как бы он не проявился, по окончании интервью хотел бы работать в вашей компании. Будет хорошо, чтобы даже те люди, кто показывает себя очень слабым, желали бы присоединиться к вам и считали это прекрасной возможностью.

В конце упомяну о том, чтобы избегать инициации или децовщины. Это происходит редко, но приводит к самым ужасным историям про интервью. В этом случае проводящий интервью берет на себя роль того, кто проводит ритуал принятия в группу. Ужасно, когда такое происходит, и вам, как нанимающему руководителю, лучше никогда не применять такого.

Окей. Это были мои пункты. Я бы хотел подчеркнуть, что я не предлагаю понижать планку для тех, кого вы нанимаете. Я полагаю, что если вы последуете этому совету, то получите более точный сигнал. Затем вы можете установить планку найма на любой высоте относительно этого сигнала, но в любом случае вы будете принимать лучшие решения о найме.

Резюмирую:

- Я рекомендую, чтобы первым шагом вы решили, какие навыки имеют значение для вашей компании. Убедитесь, что вы просматриваете кандидатов в отношении того, о чем вы действительно заботитесь, а затем разработайте структурированное интервью вокруг этих навыков.
- Соответственно, продумайте способы оценки каждого навыка, а также структурированные критерии для проводящих интервью, чтобы на них с меньшей вероятностью влияли внешние факторы.
- Затем используйте на интервью хорошие вопросы: состоящие из нескольких частей, без новых инсайтов.
- Скрывайте регалии от технических интервьюеров, потому что это вводит "шум" в процесс.
- Вам стоит подумать о цене ошибочного негатива, также как и ошибочного позитива в ваших интервью.
- И вам обычно стоит настраивать оценку на максимальные навыки каждого кандидата.
- Во время всего процесса, имеет смысл давать кандидату положительный опыт.

Вот мои советы. До этих пор, как полагаю, они были применимы как к большим, так и к малым компаниям. Теперь я пройду по некоторым пунктам, которые, я полагаю, применимы конкретно, скажем, к компаниям в рамках раунда А и малым компаниям.

Один из вопросов здесь в том, что, возможно, вы слишком малы, и не масштабировались настолько, чтобы стандартизировать ваши процессы. Если вы стартап на стадии посевного финансирования, вы нанимаете людей, вы проводите интервью с вашими первыми кандидатами, вы, очевидно, не можете использовать

очень стандартизированный процесс, потому что лишь первые несколько кандидатов услышат ваши вопросы. Это совершенно реальный аргумент. Тем не менее, я думаю, что все же стоит попробовать структурированные интервью. Ваш план скорее всего будет похож на документ google, в котором зафиксированы некоторые заметки. Однако будет полезно подумать о том, какие навыки имеют значение, и попытаться разработать вопросы для оценки этих навыков, что все равно уменьшит предвзятость в процессе интервью.

Например, возможно, у вас не выходит получать кандидатов. Вы - стартап на ранней стадии, и ваша проблема номер один в том, как получить достаточное количество знающих кандидатов для вашей компании. В этом сценарии, очевидно, ошибочный негатив стоит дороже. Соответственно, если вам трудно получить заявки от кандидатов, отбросить кого-то, кто может оказаться подходящим - это очень дорого, дороже, чем для крупной компании. Но с другой стороны, если вы маленькая компания, найм не того человека может так же дорого обойтись. Соответственно, соотношение между этими двумя стоимостями сильно не меняется, и вам придется серьезно заботиться о каждой из них. Полагаю, что вам можно быть менее агрессивным в том, чтобы рано отбрасывать людей.

Итак, если вы - компания на ранней стадии, имеющая трудности получения кандидатов, я рекомендую вам быть менее агрессивными в отборе до момента очного интервью. Допускайте больше людей и примите возможность более низкого процента успеха финального интервью, получив взамен более качественный просмотр всех кандидатов.

Пускай, например, вы - небольшой стартап, и вы еще не знаете, какие навыки имеют значение. Так что вы не уверены, нужно ли вам нанимать кого-то, кто сфокусирован на классических приложениях или на web-технологиях. Здесь нет простого ответа, и мы видим много примеров миллиардных компаний, которые идут различными путями в этом отношении.

Мой личный совет, исходя из Triplebyte и из Socialcam, и из Twitch, когда они были маленькими. Я уверен в том, что когда компания маленькая, по-настоящему важными навыками для первых нескольких наймов сотрудников являются производительность и ответственность. А именно - способность взять проект, выяснить, что нужно построить, и просто сделать это. И я рекомендую сосредоточиться на этом в ущерб качеству кода. Думаю, что первые несколько программистов, которых вам стоит принять, могут писать грязный код, но обязательно писать его быстро и доводить до конца. По крайней мере, это было так во всех трех компаниях, где я работал. Думаю, это нормально и может быть даже хорошо.

Далее, пускай вы нанимаете в области, где лично вы не имеете технических знаний. Например, вы - веб-разработчик, а хотите нанять iOS-разработчика. Что вам делать? Один из вариантов - использовать нас, Triplebyte, мы поможем вам в этом. Вы можете позвать друзей провести интервью, если такие у вас есть. Но я думаю, что неплохой

вариант - трюк, который я раньше использовал - это просто попросить кандидатов объяснить вам, как все устроено.

К примеру, вы - веб-разработчик, и вы спрашиваете iOS-разработчика о чем-то, он отвечает, и вы не знаете, хороший это ответ или плохой. Но вы можете просто попросить его объяснить. Вы говорите: "О, это интересно! Вы можете объяснить, почему это хорошая идея?" И, в большинстве случаев, если это действительно опытный iOS-разработчик, то он сможет объяснить свой ответ так, что их поймет опытный веб-разработчик. Это не всегда так. Иногда вы встретите хороших iOS-разработчиков, но плохих коммуникаторов, и это может быть для них трудным, однако в целом это хороший прием, чтобы нанимать вне области ваших знаний.

Окей. Вот и все. Я хочу закончить тем, чтобы кое о чем вас попросить. Мы разрабатываем упражнение для проводящих интервью, которое мы используем для обучения новых интервьюеров в Triplebyte, и я бы хотел попросить вас попробовать его и дать мне знать по почте, как оно пойдет, так как мне интересно, будет ли это работать в широком контексте.

Идея в том, что проводящие интервью обычно существенно недооценивают то, насколько "зашумлены" результаты интервью, и переоценивают свою способность отличить хороших разработчиков от плохих. И частью того, чтобы стать хорошим интервьюером, является то, чтобы стать несколько более скромным и знающим свои ограничения. И упражнение разработано для того, чтобы подсветить это.

Я предлагаю вам провести условное интервью с одним из ваших коллег. Пусть ваш коллега проинтервьюирует вас, исполняя роль нанимателя. Он будет задавать вопросы, а вы - играть роль кандидата и давать ответы. Я предлагаю вам заранее сказать коллеге, что вы будете давать неправильные ответы на часть вопросов, вы будете играть роль кандидата, который делает ошибки.

Затем, в течение интервью, пусть он задает вам вопросы, а вы в половине случаев давайте плохой ответ, сыграйте неправильный ответ на вопрос, а оставшуюся половину - давайте лучший ответ, который вы можете дать. И затем, когда интервью закончится, сделайте сессию с разбором, в течение которой ваш коллега проходит по всем вашим ошибкам и дает вам обратную связь.

Интересно, что он не будет знать, какие из ответов, которые вы дали, были специально даны неправильно, а в каких вы старались сказать лучшее, что вы знаете. Если он не найдет ошибку, которую вы сделали, когда хотели дать неправильный ответ, то будет выглядеть не очень хорошо. Он будет сильно мотивирован на то, чтобы быть совершенно честным и внимательным, указывая на недостатки, увиденные в интервью. Он будет критиковать как те пункты, где вы специально сделали ошибки, так и ваши попытки дать лучшие возможные ответы. Получится очень интересно.

Мы часто делаем это упражнение в Triplebyte, и оно прекрасно показывает, как проводящие интервью расходятся во мнениях. Если вы сделаете это со своим

коллегой несколько раз, то затем поменяйтесь ролями. В результате вы не только подсветите области, в которых каждый из вас слабее, но и целый ряд областей, в которых вы имеете вполне нормальную, но сильную разницу в том, что представляет собой лучший ответ на технический вопрос.

Можете быть уверены, что получится гораздо больше несогласий, чем вы ожидаете. Мне это кажется очень инсайтным. Я бы очень хотел, чтобы вы попробовали это и дали мне знать по email, как прошло.

Замечательно. У меня все. Хардж сейчас расскажет про закрытие сделки, и затем мы оба ответим на вопросы.

Хардж: Окей, я хочу в завершение пройтись по лучшим практикам и методам, позволяющим оптимизировать ваши шансы на то, чтобы люди действительно приняли предложения, которые вы им выставите.

У меня для этого пять основных советов.

Первый - просто поймите, особенно как стартап, что скорость - это большое преимущество в найме, которое вы имеете перед более крупными компаниями. Работая через Triplebyte со стартапами и крупными компаниями, мы удивляемся, насколько часто у крупной компании могут уйти недели только лишь на то, чтобы прислать актуальные параметры конечного предложения. И это когда человек уже приближается к концу процесса поиска работы, когда он в наибольшей степени готов принимать решение, идти вперед, и знает, что он делает. Поэтому, если вы как стартап быстры на каждом шаге в вашем процессе найма, от момента первого контакта и до предложения о работе, вы повышаете шансы на закрытие сделки с кандидатом. Поэтому просто будьте быстры и отзывчивы на каждом шаге.

Второй - Аммон уже упоминал его - обучайте ваших интервьюеров хорошему "врачебному этикету" общения с кандидатами. Мы спрашиваем кандидатов и разработчиков в Triplebyte, какими были причины плохого впечатления от того, когда они ходили на очные интервью в компаниях. И все ответы группировались вокруг проводивших интервью. Топ-2 проблемы были такими: 1) интервьюеры, которые сами плохо понимали технические вопросы, которые они задавали; 2) интервьюер, который очень хотел показать какой он умный, и особенно, что он умнее, чем кандидат. Если кто-то испытал отрицательные эмоции на интервью с вами, то он вряд ли примет ваше предложение.

Третий - будьте готовы поговорить в деталях о корпоративной культуре вашей компании, потому что почти каждый кандидат задаст вам вопрос: "Каково это - работать здесь? На что похожа культура вашей компании?" Думаю, мы видим особенный рост популярности такого вопроса в последний год. Честно говоря, множество обсуждений вокруг Uber и большой фокус на культуру в технологических компаниях Силиконовой долины заставляет многих, особенно частных лиц, думать больше про то, в какой культуре они хотят работать.

Здесь позвольте дать пару конкретных советов. Я бы точно подготовился к ответу на вопрос про то, как мы относимся к разнообразию, является ли ваша команда разнообразной или нет, я полагаю, что вам следует иметь некоторые мысли о том, как вы планируете включить это в практику найма.

Второе - когда вы описываете культуру, имейте в виду, что почти все компании, или их значительное большинство, используют по сути всего три прилагательных: открытая, прозрачная и сотрудничающая. Мы в Triplebyte видим это, так как мы видим, как компании создают профили. Это может быть эффективной стратегией, так как все три эти вещи - хорошие вдохновляющие качества. Не найдешь человека, который бы не хотел работать в открытом, прозрачном и коллаборативном месте. Но это не так уж хорошо для того, чтобы отличаться от других, так?

Одна из стратегий, которую вы можете применить, рассказывая о своей культуре, - это пойти на большой риск. Примером риска может быть рассказ о противоречиях, имеющихся в вашей культуре. Например, если вы верите, что у вас открытая культура, расскажите о связанных с этим противоречиях. Открытость означает быть честным во всем, а быть честным во всем может быть трудным и некомфортным для людей. Если вы пойдете по такому пути, вы можете оттолкнуть кого-то из кандидатов, но вы также можете реально увеличить процент закрытия сделок за счет кандидатов, которые больше всего впечатлены вами.

Четвертый - вовлекайте команду и инвесторов в процесс. Так, если вы сделали предложение, убедитесь, что команда затем связалась с кандидатом, предложила встретиться с ним, ответила на его вопросы и вообще вовлекалась в процесс принятия решения. И то же самое сделайте с инвесторами, специально выбирая тех из инвесторов, которые особенно хороши в контрактации с кандидатом. Например, если кандидат реально колеблется, думая, стоит ли ему покинуть крупную компанию ради вашего стартапа, возможно у вас есть инвестор, который успешно покинул крупную компанию и присоединился к успешному стартапу?

Наконец, предоставляйте кандидатам полные и прозрачные предложения, содержащие все детали, которые нужны кандидату, чтобы понять, как и сколько он получит. Это опять звучит довольно очевидно. Но я был удивлен, когда увидел, сколько компаний участвуют в такого рода гипотетической борьбе, в которой вместо того, чтобы просто сказать кандидату: "Вот такую зарплату и акции вы получите", приводит вопросы типа "Насколько вы увлечены идеей работать у нас?", "Если бы мы сделали вам предложение, как вы думаете, вы бы присоединились?" Это всегда оставляет плохое, ужасное впечатление у кандидата. Это также является нечестным, так как независимо от того, насколько сильна ваша миссия, компенсация является важным фактором при устройстве на работу. Поэтому предложение дать обязательства до того, как мы предоставим им детальную информацию ставят их в некомфортное положение и снижают шансы на принятие ими вашего предложения. Когда вы делаете предложение, предоставьте все конкретные сведения.

На этом слайде показан, я бы сказал, минимальный состав сведений, которые вам следует размещать в вашем письме с предложением. Зарплата - это и так понятно. Что касается акций - приведите детали о предложении акций, такие как количество акций в опционах, цену покупки акций и т.д. Также упомяните, что вы спрашиваете кандидата о том, насколько ему комфортно такое предложение акций. Если это старший разработчик, работавший в 6 или 7 разных стартапах, то он поймет это, не стоит уходить во множество деталей. Если же он большую часть жизни работал в Intel, то может быть не знаком с тем, как работают акции стартапа. Поэтому спросите его, насколько комфортны ему эти условия, не перегружайте его, когда представляете детали, которые кандидату не знакомы. Просто пошлите ему ссылки на вспомогательные ресурсы, которые он может почитать в удобное ему время.

Также, хотя это может звучать очевидно, убедитесь, что вы, как основатель, хорошо понимаете, как работают опционы на акции, потому что люди будут задавать вам вопросы об этом, вам не потребуется много времени на это, так как вы и сами ограничены в инструментах, связанных с акциями, и фандрайзинг обычно не включает опционы. Однако вы должны понимать, даже до уровня, например, налоговых особенностей, различные типы опционов.

Заключительная вещь, которую я собираюсь затронуть в отношении закрытия сделки - это общий вопрос, который встает перед стартапами: как вы можете конкурировать за разработчиков, особенно с такими гигантами, как Google или Facebook. Конечно, Google и Facebook могут предложить гораздо большие гибкие компенсационные пакеты, чем стартап, и честно говоря, акции публичных технологических компаний взлетают последние несколько лет, поэтому конкурировать с ними становится все сложнее. Однако, мы постоянно видим, как стартапы делают это, и побеждают в конкуренции с гораздо более крупными компаниями, предлагающими большие деньги.

Я полагаю, что есть четыре посыла, которые они озвучивают, чтобы быть очень конкурентоспособными. Первый подчеркивает обучение. Речь идет о том, что вы учитесь больше и быстрее всего, когда на вас лежит реальная ответственность за принятие решений, это означает, что если вы ошибетесь, случатся действительно плохие вещи. И шанс, что такое может происходить в крупных компаниях, очень-очень мал. Большие компании имеют систему сдержек и противовесов, которая не дает никому принести слишком большой ущерб, а особенно новичкам. Поэтому вы говорите кому-нибудь: "Эй, у нас нет такого уж люкса. Мы быстро растем, у нас все надо было выполнить еще вчера, и ты сразу погружаешься прямо в процесс принятия настоящих решений со всеми полномочиями, и, если ты облажаешься, это будет плохо для всей компании. И это тот путь, который позволит тебе научиться и получить реальный опыт." Итак, подчеркивайте это.

Второй посыл - это про быструю карьеру. Один из самых общих комментариев, которые я говорю кандидату, такой: если посмотреть на топ-менеджеров крупных публичных технологических компаний, они могут быть гораздо моложе, чем такие же топ-менеджеры в публичных производственных или сервисных компаниях, не правда ли? Причина того, что ты можешь профессионально вырасти и стать директором по

продукту Facebook в 38, или сколько там этому СРО, состоит в том, что ты присоединился к стартапу на самом раннем этапе и рос вместе с ним, так? Технологическая отрасль - единственная, где это происходит именно так. Это реально меритократия, вы не обязаны платить компании 20-ю годами вашей жизни, прежде чем она рассмотрит вас на старшую позицию, вы можете вырасти настолько быстро, насколько растет стартап. Итак, я бы подчеркнул это.

Третий посыл - говорите о стоимости альтернативы. Конкретно я рассказываю об этом в рамках того, что в любой момент времени всегда есть набор крупных безопасных технологических компаний, в которых можно работать. Их названия могут меняться примерно каждые пять лет, но опыт работы в них остается примерно одним и тем же. В то время как опыт работ в стартапах - очень разнообразный. Я бы подчеркнул тот факт, что ваш стартап и команда прямо сейчас составляют уникальную возможность, прекрасно подходящую кандидатуру, которую он в дальнейшем не получит. А если это не сработает, он всегда может перейти в обычную крупную компанию.

Последний пункт. Есть одна особенность в найме более молодых кандидатов. Расскажите про наставничество. Одна из вещей, которую мы особенно видим у выпускников колледжей, - они беспокоятся о том, что не смогут получить достаточное наставничество, что они не научатся лучшим практикам, присоединившись к стартапам. И поэтому может быть им стоит пойти работать в Google на пару лет и, получив такой опыт, затем устраиваться в стартап.

Во-первых, вам следует решить, возможно, что это правда. Если в вашей команде только молодые разработчики, и, по-честному, вы просто не хотели бы заниматься наставничеством и выводить кого-то на высокий уровень, то не делайте таких предложений и, наверное, не нанимайте таких людей. Но если в вашей команде есть опытные разработчики, подчеркните тот факт, что новички будут учиться, работая бок о бок с опытными разработчиками, которые работали в разных местах и обладают лучшими практиками и т.п.

Ну вот и все. Думаю, мы готовы отвечать на вопросы. Мы ответим на вопросы по темам, о которых мы говорили.

Голос из зала 1: Это обсуждение больше про разработчиков и технологии, но могли бы вы рассказать о таких вещах, как проверка опыта, которые вычеркивают кандидатов из рассмотрения?

Хардж: Что вычеркивает кандидатов?

Голос из зала 1: Стоит ли применять проверку опыта?

Хардж: Проверка опыта. Честно говоря, большинство стартапов, которые я знаю, не делают таких проверок. Однако сейчас это очень просто сделать, потому что можно интегрировать Checker с сервисами расчета зарплаты, поэтому это точно возможно, но

я бы не фокусировался на этом. Я бы сосредотачивался на том, подойдет ли человек компании или нет.

Голос из зала 2: Я знакома с молодым специалистом по специальности информатика, который хочет стать интерном у меня. У меня нет разработчиков в команде... [неразборчиво], так как мне надо всего лишь... [неразборчиво] Что мне делать?

Аммон: Это непростой вопрос. Большая часть молодых специалистов по информатике обычно не смогут хорошо выполнять работу, которую вы им поручаете. Если иметь это в виду, то в данном случае вы могли бы просто оценить работу, которую человек делал в прошлом. Соответственно, вы могли бы попросить человека прислать вам портфолио либо другие примеры работы, которую он выполнял, и, если она будет соответствовать уровню и виду, которым вы будете довольны, то, возможно, принять его на работу.

Голос из зала 3: Есть ли методика, которую вы действительно могли бы использовать, чтобы помочь принять правильное решение о найме людей, типа Майерса-Бриггса или... [неразборчиво]?

Аммон: Это возможно. То, на чем мы фокусируемся, это найм, "слепой" в отношении прошлого опыта, таким образом мы сосредотачиваемся на прямой оценке навыков и не смотрим на другие вещи, которые коррелируют и предсказывают результат. Типы личности реальны, и люди сильно различаются по ним, и это важно для выполнения работы, и я полагаю, что есть и другие исследования про 5 больших типов личности, а не только Майерса-Бриггса. Всё это имеет значение, однако я бы постарался точно делать это отдельно от технической оценки. Поэтому - оцените технические показатели, а затем отдельно попытайтесь оценить культуру, дружелюбие, софт-навыки. И потом объедините все результаты и постарайтесь принять общее решение.

Голос из зала 4: После того, как [неразборчиво]. Я хочу найти правильного технического специалиста, и я знаю, что вы, ребята, используете различные инструменты и способы, чтобы получить разработчика в команду и начать давать ему задачи. Но, если я не технарь, как мне оценить, насколько человек компетентен, если я не имею опыта в этом?

Аммон: В качестве крайнего ответа я бы сказал, что, возможно, никак. Итак, вопрос - если вы основатель, не являющийся технарем, как вам оценить технические навыки человека? Я думаю, что ответ, к сожалению, что, возможно, вы не можете этого сделать. Вы также можете воспользоваться советом о том, чтобы посмотреть прошлую работу, но я считаю, что причина того, что считается очень полезным иметь сооснователем технаря, состоит в том, что иначе вы находитесь в фундаментально не выигрышном положении.

Голос из зала 4: Я имею в виду, что поскольку поиск кандидата... [неразборчиво]

Аммон: Да, я... Ты что-то думаешь по этому поводу?

Хардж: Вопрос - как найти технического сооснователя и понять, хорош он или нет? Я полагаю, что совет Аммона здесь подходит. Используйте друга или того, кто является разработчиком, чтобы поговорить с кандидатом и выяснить, является ли кандидат хорошим разработчиком или нет. Других вариантов нет.

Голос из зала 5: Вопрос про генерацию талантов, как шаг перед получением кандидатов. Есть ли какие-то методы, связанные с работой с университетами, и какие при этом возникают вызовы? Или есть ли такой механизм получения кандидатов, который может быть хорошим способом растить таланты, а не брать со стороны?

Хардж: Да, но мне кажется, что вопрос в большей степени касается... О, извините... Повторяю вопрос. Вопрос - есть ли способы работать с университетами и колледжами, чтобы генерировать как бы трубопровод талантов, откуда вы могли бы их нанимать, я правильно сформулировал? Я бы сказал да, такие способы точно есть, например карьерные ярмарки и спонсорские мероприятия в колледжах и т.п. Но я полагаю, что эти способы лучше работают для больших компаний, которые могут подстраиваться под сроки обучения и расписание колледжей. Для стартапов это довольно трудно, потому что обычно вам требуется кто-то, кто доступен прямо сейчас для найма, и вы не можете ждать, когда он закончит образование. Поэтому я бы не советовал стартапу вкладывать слишком много времени в это.

Аммон: Для прояснения, крупные компании делают предложения за 4 года вперед с большими бонусами записавшимся, предоставляемыми заранее, поэтому для стартапов очень сложно конкурировать с наймом в колледжах.

Голос из зала 6: Вы говорили про ролевую игру в плохого интервьюера. Мне интересно, почему вы говорите про плохого интервьюера, а не про [неразборчиво] который делает ошибки.

Аммон: Да. Вопрос на прояснение моего последнего пункта про ролевую игру с плохим интервьюируемым. Ключевой момент здесь в том, что вы играете интервьюера, а коллега, который играет кандидата и дает ответы, будет намеренно делать ошибки. Это позволяет его коллеге быть критиком.

Обычно, когда вы моделируете интервью между двумя коллегами, проблема состоит в том, что каждый будет "имитировать удары". Они не будут честно высказывать свое мнение об ответах своего коллеги. Это просто не работает из-за социального давления. Если же вы скажете ему: "Я буду специально делать некоторые ошибки," это дает ему свободу потом указывать на недостатки в вашем интервью. И он не только может это сделать, но и будет должен это сделать, потому что вы сказали ему: "Я делаю ошибки", и вы спросили его "Каково твое мнение о моих результатах?" Если он не укажет на ошибки, которые вы сделали, то получается, что он что-то пропустил, так?

То есть такой подход переворачивает социальное давление и создает давление в том, что он будет мотивирован на критику, что очень полезно, так как в качестве опытного разработчика старшего разработчика в команде успешной компании вы давно не получали реально жесткой критики того, насколько вы хороши в ответах на собственные вопросы.

Произойдет то, что они укажут на вполне законные пробелы. Вы будете пытаться дать лучший свой ответ, а они укажут то, что вы сказали неправильно или то, что могло быть сказано лучше, и это очень способствует скромности. И я рекомендую это упражнение, в котором вы можете представить себе такой опыт скромности и опыт того, когда тебе указывают на проблемы в том, что ты делаешь.

Джефф: Окей, из-за ограничений по времени, прошу задать еще один вопрос, пожалуйста.

Голос из зала 7: На одном из слайдов вы говорили про 95% успеха при найме, однако принимая во внимание то, что те, кто проводит интервью обычно не поддаются надежной рейтинговой оценке, трудно поверить, что там действительно 95%... Я имею в виду, что это также не совпадает с моим опытом, что 95% наймов были бы успешными. В связи с этим, что вы думаете о реальном проценте успеха, при котором через год после найма человек все еще работает в компании и является одним из лучших работников?

Аммон: Да. Число 95% взято из опросов, которые мы проводили в компаниях. Окей, извините. Вопрос - на одном из моих слайдов я сказал, что найм успешен в 95% случаев. Но это не совпадает с опытом менеджеров по подбору и с опытом найма людей. Многие люди, которых вы нанимаете, не столь уж хороши. Поэтому вопрос в том, каков реальный процент найма лучших работников?

И да, вопрос очень правильный. Этот процент взят из опросов компаний, которые мы делали, в которых мы спрашивали компании: из всех нанятых ими людей сколько процентов сотрудников было уволено? И мы действительно спрашивали о проценте лучших работников. И 5% - это процент уволенных. Поэтому 5% действительно увольняют, а еще 30% - это люди, которые остаются, но не являются по настоящему прекрасными работниками. И около 10% относятся к той категории, которых компании называли своими самыми лучшими приобретенияами и прекрасными работниками.

Джефф: Окей, спасибо ребята!

Хардж: Спасибо!