**Вы достигли высот самообучения. Что дальше?**

***9 эффективных рекомендаций, которые не стоит откладывать на потом.***
Возможно, вы увлечены изучением Boot Camp либо проходите 100-дневную программу освоения программирования. Независимо от того, чем из подобных вещей вы занимаетесь, однажды настанет день, когда вы достигните определенной точки развития и сможете назвать себя разработчиком-самоучкой.
После этого последует простой вопрос: “А что дальше?”
Всегда найдется тот, кто разбирается в этом вопросе лучше вас и может программировать более эффективно.
Как же понять, что вы уже достигли того уровня развития, чтобы уверенно назвать себя полноценным разработчиком? Это тяжело, особенно если вы следуете по пути самообучения. Те, кто идет по альтернативному пути получения полноценного компьютерного образования, будут всегда на шаг впереди вас. Так происходит хотя бы потому, что у них есть то, чего нет у вас: как минимум 3 года упорного изучения внутреннего функционирования компьютерных программ.
Эта сфера распространяется от алгоритмов до различных теорий, что по определению и делает их лучшими разработчиками, чем вы.
Но фактически эффективный и практичный разработчик выявляется не только по наличию научной степени, равно как и не всегда по количеству лет, проведенных в этой отрасли.
Каждый разработчик сталкивается с разной степени сложности решаемых рабочих задач, что в итоге и определяет его скорость и качество выполнения работы. Кодовая интуиция нарабатывается посредством опыта, а именно количеством проработанных ошибок. Это именно так, потому что, когда все легко складывается — мы не учимся по-настоящему.
*Реальное* освоение приходит только тогда, когда мы решаем *реальные* проблемы.
Вот 9 методов, систематически следуя которым, вы расширите сознание, наберетесь опыта и усовершенствуете практические навыки.

**1. Создавайте что-нибудь**
Будучи разработчиком, вы словно бизнесмен, создающий коды. Поэтому лучшей практикой будет выполнение тех заданий, которые вам потребуется делать регулярно. Как бы то ни было, существует два варианта разработки: следование чьему-то конкретному указанию либо проектирование кода согласно своей собственной креативности.
Второй способ подразумевает создание собственных задач.
Все это делается лишь для того, чтобы в итоге вы могли создавать код на ходу, с минимальным отвлечением на вспомогательные ресурсы типа Stack Overflow. Ваш богатый опыт позволит применять все полученные ранее кусочки знаний вместе гораздо быстрее и не тратить кучу времени на поиски ответов.
Как только вы достигните этого уровня, вам станет легче просто фокусироваться на процессе создания продукта, на его архитектуре, а не на том, где и какие взять для этого инструменты.

**2. Изучайте новые фреймворки или библиотеки**
Мир разработки никогда не стоит на месте и уже не достаточно просто знать язык программирования. Вам необходимо изучать различные фреймворки и библиотеки, разбираясь до тех пор, пока вы не сможете соответствовать их правилам, синтаксическим и другим требованиям, которые будут иметь значение в перспективе.
Их изучение также дает понять потенциальным работодателям, что вы легко усваиваете новые знания. Особенно это касается организационных стратегий написания кода, т.к. по сути библиотеки и фреймворки ими и являются.
Они существуют для облегчения процесса разработки с помощью преобразования сырого кода в структурированную методологию по созданию приложений, которая основывается на наборе конкретных требований. Они также помогают держать вам все под контролем на одной странице. Особенно это полезно, когда приложение уже достигает определенного размера, и над ним трудится целая команда разработчиков.

**3. Углубляйтесь в научную компьютерную тематику**
Если вы самоучка, то скорее всего во многом не дотягиваете до обладателей профессионального компьютерного образования. Основная разница между вами и людьми, имеющими высшее образование, — это их знание основополагающих вычислительных идеологий, которые не разъясняются в стандартных программах по самообучению.
В большинстве случаев самоучки сразу приступают к написанию кода, не вникая в нюансы, касающиеся рекурсии, алгоритмов, функционального или объектно-ориентированного программирования.
Итоговая рабочая программа скорее всего будет страдать от недостатка устойчивости и лаконичности кода из-за отсутствия фундаментального метазнания.

**4. Научитесь правильно мыслить**
Независимо от того, какой путь вы выбираете, правильный образ мышления является жизненно необходимым компонентом в вашем становлении высококлассным разработчиком. Я сейчас не имею в виду ваше стандартное мышление, a скорее вашу способность посмотреть на задачу под определенным углом прежде, чем решать ее.
Существуют различные уровни мышления, начиная с уровня кода, через архитектурный уровень и до концептуального. Различные парадигмы, методологии, фреймфорки и библиотеки — все эти составляющие процесса имеют свои особенности, которые вынуждают смотреть на задачу под определенным ракурсом в зависимости от ее конкретного уровня.
Так что расширяйте ваше сознание, изучая все аспекты программирования.
Старайтесь находить ту точку, где можно сделать объективное разделение и установить связи между различными частями задачи для нахождения особого пути ее решения. При таком подходе у вас будет получаться креативный высоконадежный код, обладающий достаточной гибкостью для дальнейшего удобного изменения.

**5. Отправляйтесь за покупками**
Электронная коммерция составляет большую часть рынка разработки софта. Вы можете не только анализировать некие полученные данные, но и выяснять, как можно применить конкретный опыт покупок.
Как вы думаете, почему платформа Shopify так развита? Все дело в ее тотальной проработке покупательского цикла от и до. Вы, как разработчик, можете позаимствовать у этой платформы очень много полезного для своих проектов. Бесплатный аккаунт поможет вам разобраться с непонятными транзакциями и интеграцией.
Но Shopify далеко не единственный полезный сервис. Взгляните, к примеру, на поток посетителей Amazon. Какие стратегии применяет этот гигант торговли? Какие вы можете отследить модели или техники, успешно применяемые и на других платформах?
Покупатели охотнее совершают покупку, когда встречаются со знакомой моделью сервиса, потому что в таких случаях они уже знают, как все работает. Вам нужно лишь выяснить, что это за модели, и затем просто использовать их в ваших проектах, связанных с электронной коммерцией.

**6. Инвестируйте в хобби**
Хобби расширяют нашу сферу деятельности и способны сделать из нас экспертов в совершенно различных областях. Они особенно полезны, когда нам требуется найти некое нестандартное решение извне.
На большинстве университетских факультетов помимо основного предмета изучения, присутствует общий образовательный курс, расширяющий потенциальные возможности выпускника в будущем.
Того же можно добиться и посредством хобби, которое может помочь раскрыть альтернативные методики решения некоторых задач. Хобби также может дать вам преимущество перед другими кандидатами на трудоустройство, если выяснится, что оно соответствует специфики деятельности компании — работодателя.
Например, тот, кто увлечен тематикой здорового образа жизни очень удачно впишется в компанию, которая работает в секторе здравоохранения. Все потому, что кандидат уже будет разбираться во многих соответствующих темах, которые для других окажутся малознакомыми.

**7. Выходите из своей зоны комфорта**
Когда дело доходит до разработки приложений, мы часто прибегаем к пути наименьшего сопротивления для того, чтобы успеть все сделать вовремя. Тем не менее цель не всегда оправдывает средства.
Как только мы привыкаем выбирать простой путь, необходимо давать себе встряску, которая поможет нам расти как разработчикам.
Где же получить такую встряску? Выйти из зоны комфорта, что буквально означает заняться теми областями знаний, которые нам кажутся сложными. Это может быть некая парадигма или методология, “конкурентный” фреймворк или библиотека, изучение которых вы все время откладывали.
Как это перебороть? Иногда даже простой взгляд на то, как этим занимаются другие, вселяет в человека необходимое стремление осваивать что-то новое и двигаться дальше.
Написание кода, на одну половину, —  творчество, а на вторую — индустрия знания. Успех без любой из этих половин невозможен.

**8. Стремитесь к работе своей мечты**
Когда вы в последний раз искали объявление о работе своей мечты?
Работодатели часто указывают список требований к кандидатам, особенно когда речь идет о совмещении различных технологических навыков.
Велика вероятность того, что вы не сможете подойти по всем пунктам требований, но соответствие большему их числу по сравнению с конкурентами повышает ваши шансы на трудоустройство в желаемое место.
Работа требует знания React, а вы работаете в Angular? Значит пришло время пойти и поиграть в песочнице, чтобы ознакомиться и с этим инструментом. В требованиях указана какая-то странная аббревиатура, о которой вы даже и не слышали? Похоже, что пора ее прогуглить.
Когда вы изучаете потенциальные вакансии, не обязательно подходящие вам сейчас, а те, которые вы займете, став сеньором — это формирует в вашем сознании некий оттиск всех необходимых для этого навыков.

**9. Освойте искусство тестирования**
Тестирование обычно не находится в поле для самообучения. Изучение тестирования фреймфорков и библиотек активирует в вас особый способ мышления и отточит навык написания кода, в том числе.
Что касается принципов работы тестирования, то они полностью перевернут ваше представление о вещах и развернут процесс написания кода в профиль. Вы начинаете с того, что понимаете, какую конкретную форму должны иметь данные, и тестируете, проверяя вводные характеристики, а также ожидаемый результат.
Это превращает процесс написания кода из беспорядочного тыкания в неизвестность в процесс разработки через тестирование, в котором форма каждой детали гармонично вписана в единую архитектуру всего кода. Этот навык также выдвинет вас на шаг вперед перед другими кандидатами на желаемую вакансию в будущем. А все потому, что, скорее всего, они не заглядывали в раздел тестирования совсем.

**Заключение**
Не существует линейного пути развития ваших навыков в качестве разработчиков. Однако существуют некоторые рецепты и комбинации, которые могут существенно расширить границы ваших текущих знаний и опыта программирования.
Основная особенность самообучения — это его повышенная сложность ввиду недостатка руководства. Но если вы уже зашли далеко в этом своем путешествии, то для продвижения потребуется лишь продолжать делать уверенные шаги в этом направлении.
Не имеет значения, в каком направлении вы идете. Пока вы продолжаете учиться — вы продолжаете развиваться так или иначе.
Никогда не прекращайте учиться, т.к. это ваш решающий рычаг для достижения успеха в сфере, где он зависит от того, как много вы знаете и умеете.
Благодарю за внимание!