Теперь, когда вы уже имеете представление о веб-разработке, попробуем ответить на вопрос: а как, собственно, в нее влиться?

Относительно простыми способами попасть в этот мир, можно назвать следующие:

* Устроиться ручным тестировщиком
* Научиться верстать

Все остальные способы требуют умения программировать. Знания необходимые веб-разработчику можно разделить на несколько направлений.

Информатика

Как я уже неоднократно упоминал, в разных уроках и курсах (например в "Жизнь программиста"), существует набор вещей, которые нужно знать независимо от выбранного в дальнейшем направления:

* Элементы дискретной математики
  + Графы
  + Автоматы
  + Кодирование
* Алгоритмы и структуры данных
* Теория множеств
* Логика
* Системы счисления

В хороших местах смотрят именно на вашу базу, потому что довольно глупо от начинающего разработчика требовать знания прикладных инструментов. Иначе это уже не новичок. Не забывайте что в такие компании как Google, Facebook или Яндекс попадают прямо со студенческой скамьи. При том, что решаемые задачи гораздо более сложные, чем разработка сайтов.

Пусть вас не пугает то, что было перечислено выше, в реальности уровень владения, который понадобится, очень примитивен. Более того, вы уже, скорее всего, знаете многое из того что описано выше, но не сможете это сформулировать. Все же мы 10 лет учились в школе, а математики и физики там было хоть отбавляй.

Отдельной темой являются разные виды мышления:

* Логическое мышление
* Абстрактное мышление
* Системное мышление
* Вычислительное мышление

Мышление решает все. Именно его уровень и определяет то, какие вы будете выдавать решения. Прокачка мышления, на ранних стадиях, намного важнее прикладных и статических знаний. Хотя это сложно увидеть, так как нельзя пощупать, но именно мышление определяет то насколько сложным или простым будет решение, насколько поддерживаемым, расширяемым и дешевым.

Хекслет делает акцент именно на этом. Первые курсы целиком и полностью построены так, что происходят постоянные переосмысления и изменения в том, как вы думаете о стоящих перед вами задачах и коде. Это неоднократно отмечено большинством наших пользователей, а так же компаниями, которые собеседуют наших выпускников.

Специальные знания

Ниже перечислен список технологий, которые важно знать, хотя бы на базовом уровне любому веб-разработчику для устройства на работу. С ростом профессионального уровня глубина будет увеличиваться, но пока вы только начинаете свой путь, достаточно иметь общее представление. В любом случае, после прохождения первых собеседований, станет понятно, что из перечисленного ниже желательно дотянуть.

Клиентская часть

* HTML
* CSS
* JavaScript

Софт

* Популярные веб-сервера и их принципы работы
* Git и системы контроля версий в принципе
* SSH (+ работа с ключами)

Операционные системы

* Уверенное владение командной строкой Linux
* Базовое администрирование Linux
* Процессы/Потоки
* Сеть

Операционные системы это одна из наиважнейших тем в программировании. У разработчиков постоянно возникают сложности и вопросы, большая часть ответов на которые, к их удивлению, лежит в области знаний операционных систем.

Сеть

* HTTP
* DNS
* TCP/IP

Стек (для любого языка)

Будет полезно знать, но на работу, скорее всего, возьмут и без этого:

* Микрофреймворк
* Роутинг
* ORM
* Шаблонизация

Пожелания от Хекслета

* Автоматизированное тестирование
* Управление конфигурацией (Ansible и подобные)
* Процессы (Agile: Scrum, Kanban)
* Теория ограничений
* Lean Startup

Тесты

То, про что обычно не говорят и очень часто не используют. При этом невозможно себе представить профессионального разработчика, который бы не писал тестов. Такие люди конечно же есть, но их нельзя называть профессионалами (они так не считают).

Все остальное, это то с чего, например, мы обучаем программистов, работающих в Хекслете. Ведь программист, это не тот кто должен писать код, это человек решающий задачи бизнеса, как правило, с помощью написания кода, но еще лучше если он будет решать их с помощью удаления кода. [Подробнее](https://ru.hexlet.io/blog/posts/developers-business-value)

Книги

В течении своей профессиональной карьеры, я собирал лучшие из лучших книг, тех чьи знания не устаревают со временем и таких, которые важны независимо от вашего рабочего окружения. Список доступен по [ссылке](https://map.hexlet.io/pages/books).

Часть из этих книг читается очень легко, часть требует глубокого погружения. Подробно вопрос о том "как читать книги" разбирается в [этой статье](https://ru.hexlet.io/blog/posts/how-to-read-books).

Хекслет

Необходимо сделать ремарку. Хекслет, в текущем состоянии, ориентирован на обучение программистов. У нас нет курсов по CSS и HTML и мы исходим из того, что вы их либо знаете, либо изучаете параллельно.

Более того, в наших курсах, мы довольно долго идем к непосредственно веб-разработке. Это связано с тем, что перед нами стоит амбициозная цель не просто дать вам прикладные знания по конкретному набору технологий, а сформировать правильное мышление, привить культуру кода и подготовить для работы над, по настоящему, серьезными проектами. Состояние кипящего мозга в процессе решения задач, на Хекслете, это нормальное состояние -).

А что насчет требований в вакансиях?

Лучше всего, в вакансиях для новичков, к требованиям знания конкретных технологий, библиотек и даже языков, относиться как к "мы используем в работе", другими словами ничего страшного в том, что вы не знаете этих вещей. Посмотреть и почитать про них будет полезно, но учить точно не стоит до тех пор пока вы не поймете зачем вам оно.

Типичный пример, это знание jQuery на позицию верстальщика или frontend-разработчика. В реальности,jQuery, представляет из себя набор функций для разных задач, которые учатся по ходу действа и ровно в том объеме, который нужен для решения конкретной задачи (учить его от и до абсолютно бессмысленно). При этом, в вакансиях не пишут о, по настоящему, важных вещах. Работа, того же, jQuery целиком и полностью завязана на DOM. И вот без понимания объектной модели документа, использование jQuery приводит к совершенно неадекватному коду.

Заключение

Предвижу вопрос "Неужели так много надо знать?". В реальности, знать нужно на порядок больше. Даже зная описанное выше, в первый месяц работы программистом, вы будете каждый день узнавать огромное количество нового, а старое понимать лучше и глубже. Программирование достаточно серьезная область, в которую с нуля после пары месяцев обучения попасть нельзя. Кому-то может и везет так, но и задачи будут соответствующими.

Следующие курсы погружают вас в мир программирования. Этот путь сложен и требует большой дисциплины, желания и самостоятельной работы. Надеюсь что вы пройдете его до конца и добьетесь своей цели.