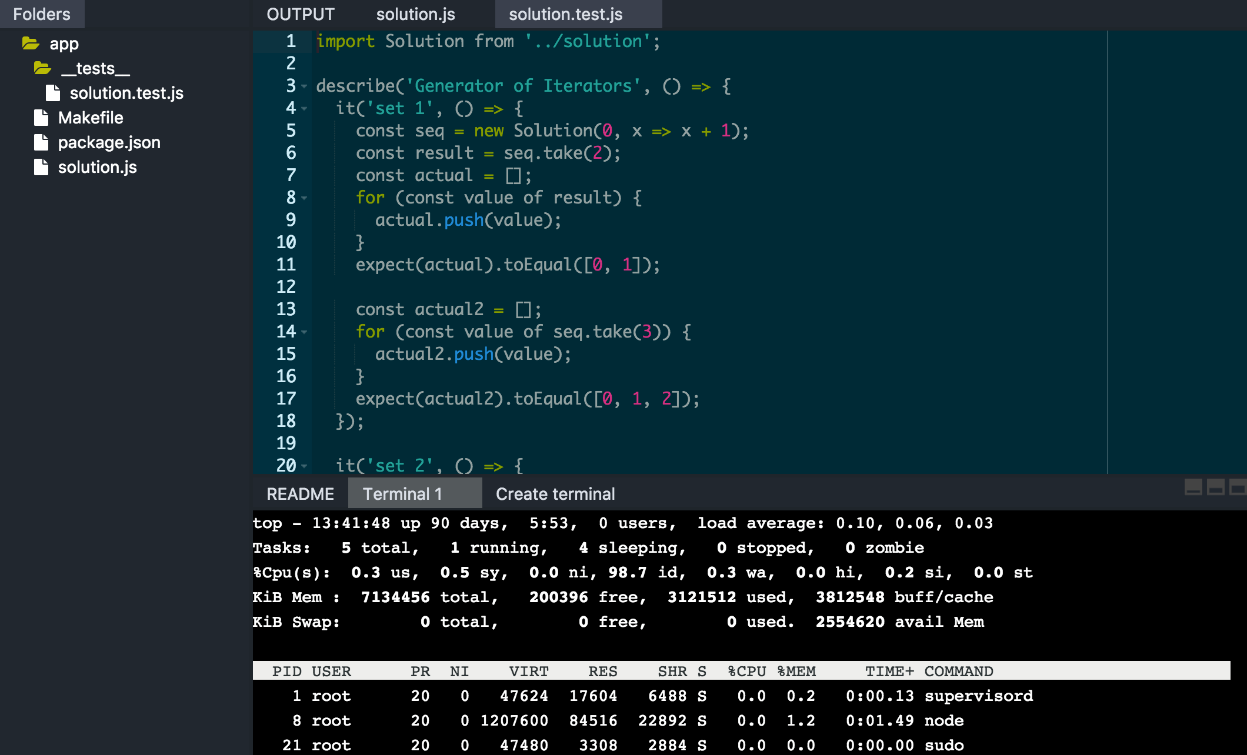
Как правило, одной бекенд разработки достаточно для полноценного существования проекта. Даже от верстки можно уйти, если использовать css-фреймворки наподобие Bootstrap. Более того, большинство существующих сайтов это как раз дизайн + верстка + бекенд. Но есть ситуации, когда такая схема не работает.

Как пример, давайте рассмотрим среду выполнения практических заданий на Хекслете. Первое, что бросается в глаза при работе с ней, это отсутствие классической веб схемы запрос-ответ. Страница никогда не перезагружается, а сам сайт, в этот момент, выглядит как полноценный редактор кода. На самом деле, это и есть полноценный редактор кода -).



Реализовать подобную интерактивность через серверную логику невозможно по определению. Ведь в таком случае, переход по любой ссылке, приводил бы к перезагрузке страницы. Согласитесь что выполнять упражнения стало бы нестерпимо больно. К счастью, из ситуации есть выход.

Уже в начале 1990х годах стало понятно, что существующих решений недостаточно для создания интерактивных (динамичных) сайтов. Компания Netscape Communications (создавшая один из первых и, в свое время, лучших браузеров) разработала специальный язык выполняющийся в браузере. Этот язык сначала назывался LiveScript, а затем был переименован в JavaScript. Он не имеет никакого отношения к языку Java, но в те года Java быстро становилась популярной и этим фактом решили воспользоваться создатели JavaScript использовав слово Java в названии языка.

В конце концов, большинство браузеров поддержали JavaScript. Этот факт имеет огромное значение. Так, JavaScript начал свое шествие став одним из самых популярных и массовых языков в истории. Вышел далеко за пределы браузеров и компьютеров как таковых. Например, JS часто используется не только как клиентский, но как и серверный язык программирования.

Общий принцип работы JavaScript в браузере следующий. Кроме HTML с сервера подгружаются JS скрипты, которые затем начинают выполняться на странице. Браузер не просто умеет исполнять JS, но так же предоставляет возможности по манипулированию как самим браузером (ограниченно) так и элементами на странице (посредством DOM API).

Благодаря этому, стало возможно создавать сайты, по возможностям не уступающие обычным приложениям. Microsoft Office для многих ушел в прошлое, а на его место встал Google Docs, даже сложнейший инструмент Photoshop уже представлен онлайн. Социальные сети наполнены разнообразными приложениями построенными на базе современных фронтенд технологий. Ну и конечно игры.

Важно понимать, что Frontend-разработка не является неотъемлемой частью веб-разработки. Реальная потребность в богатом интерфейсе возникает не на каждом втором сайте. К тому же, введение логики на фронтенд, существенно усложняет и удорожает разработку. А сайты, чей фронтенд целиком строится в браузер, называют Single Page Application (SPA).

Современная фронтенд разработка [крайне сложна](https://habrahabr.ru/post/312022/). Начиная от того что поверх JS созданы десятки языков, устраняющие некоторые из его недостатков, заканчивая огромным количеством фреймворков, серверного инструментария и направлений работы.

Нарастание сложности привело к выделению категории людей, которых называют фронтендерами. Здесь остановимся по подробнее. Часто говорят "нам нужен фронтендер". За этой фразой может скрываться многое. В какой-то момент стало не модно говорить "верстальщик" и его тоже заменили словом "фронтендер". Итого получается что под "фронтендер" имеют ввиду следующее:

1. Верстальщик.
2. Программист на JS и одновременно верстальщик.
3. Программист на JS хорошо знающий верстку, но без задач на нее.

Все три это совершенно разные люди. Я, например, неплохо знаю JS и пишу много фронтенда Хекслета, но при этом совершенно не умею верстать и более того, ни за какие коврижки не готов этим заниматься. Версткой за нас занимаются те, кто разрабатывает Twitter Bootstrap -). Верстальщики как правило знают базовый JS, но программировать не умеют. Иногда они немного работают с jquery и добавляют простые элементы, например, слайдер. С другой стороны, фронтендеры чаще всего вырастают именно из верстальщиков, но со временем ситуация меняется. Многие backend девелоперы тоже с удовольствием либо пишут, либо переходят на frontend (обратное тоже верно).

Совершенно нормальная ситуация (и я являюсь сторонником такого подхода), когда есть отдельно верстальщик и программист, который по необходимости пишет фронт или бек в зависимости от задачи. В этом нет ничего сложного, особенно учитывая то что современный фронтенд очень сильно опирается на бекенд с одной стороны, и с другой предлагает достаточно зрелые инструменты для решения типовых задач.

Главное что хочется донести до всех, кто планирует стать фронтенд-программистом, то что этот человек в первую очередь программист, а во вторую уже фронтенд.

Если говорить про инструментарий, то можно сказать, что совсем недавно[React](https://facebook.github.io/react/) произвел революцию в том как можно строить фронтенд приложения. В 2015 году, я выступал на конференции где[подробно рассказал](https://www.youtube.com/watch?v=DCeNCr2tKOI) об этом.