Прочее

- Общие требования к сотрудникам
- Тестовое задание для практикантов
- Тестовое для стажеров
- PHP junior - требования
- 9 учебных проектов для бэкендера
- Что почитать
- Программирование — это скучная магия

Общие требования к сотрудникам

В Вебтолке мы всегда ориентируемся на положительный результат в работе. Да, процесс очень важен, но в конечном счете мы работаем на результат. Это значит, что фразы : "я работал\я делал" - заранее обернутся факапом. Если ты считаешь, что тебе платят зарплату за то, что ты просто посидел на работе и поработал, то с таким отношением к работе мы долго не сможем работать вместе. Мы работаем иногда сверхурочно не потому что, это наказание за прогул\опоздания\или другую херню. Мы работаем по субботам, потому что мы не успеваем сдать проект во время и сделать положительный результат для клиента.

В Вебтолке всегда будет выслушано мнение любого сотрудника по любому вопросу. Даже если ты не руководитель, твое мнение важно, так как мы работаем в команде, и каждый зависит друг от друга. Здесь ценятся думающие люди, несмотря на должность. Думать головой нужно вообще всем, даже грузчикам, а тем более программистам. Это значит, что если ты видишь проблему или знаешь как улучшить выполняемый процесс, не стоит молчать.

Все члены команды должны чувствовать ответственность и уважение друг перед другом. Чем меньше компания, тем важнее, чтобы сотрудники принимали решения самостоятельно, быстро адаптировались и пересматривали приоритеты, и делали все от них зависящее, независимо от должности и полномочий. Даже если это означает, что руководитель должен поработать грузчиком в своей компании, бухгалтер — спуститься в магазин и помочь сформировать заказ, а генеральный директор — самостоятельно обслуживать клиентов. Когда люди говорят «Это не моя работа», значит, они думают исключительно о себе. Это быстро подрывает общую производительность, так как сплоченная команда превращается в группу отдельных индивидов.

В Вебтолке работают люди, которым изначально была интересна данная отрасль, и в которой люди хотят развиваться. Если ты чувствуешь застой в навыках, или хочешь чему то научится, стоит об этом поговорить. Возможно ты вообще хочешь заниматся чем - то

совершенно другим для компании, об этом тоже стоит поговорить. Если ты не хочешь дальше развиваться, скорее всего эта работа будет делать тебя несчастным в будущем, какой смысл работать на нелюбимой работе? Это значит что ты должен постоянно развиваться, и вправе требовать системы обучения.

Правило. Каждый специалист имеет право ошибаться. Хороший специалист делает выводы и становится лучше.
Не ошибается тот, кто ничего не делает. Важно анализировать повторные однотипные ошибки. У нас никогда не было и не будет штрафов. Обсуждение ошибок необходимо.

Процесс имеет значение

Люди и человеческие отношения не могут быть оценены одними только цифрами.

Пример 1. Менеджер завершил проект в срок.
Но в ходе работы не держал клиента в курсе дел. В итоге клиент испытывает двойственное ощущение и сомневается в продолжении сотрудничества.
В худшем случае больше не работает с нами.

Пример 2. Менеджер завершил проект в срок, в ходе работы держал клиента в курсе дел. Клиент доволен.
Но менеджер постоянно требовал, чтобы специалисты задерживались. Команда недовольна ходом проекта.
В худшем случае кто-то из специалистов уволился.

Пример 3. Дизайнер — профессионал высокого уровня с точки зрения качества работы. Но процесс нервирует коллег, или менеджерам сложно положиться на такого специалиста.
В результате портится атмосфера в коллективе.

Правило. Процесс так же важен, как и результат.

Тестовое задание для практикантов

Библиотеки с помощью которых нужно выполнить тестовое задание:

- Idiorm - https://idiorm.readthedocs.io/en/latest/ (Работа с базой данных)
- Slim - Сейчас актуальная третья версия, но можно воспользоваться второй. Она полегче для понимания http://docs.slimframework.com (роуты, аналог контроллеров)
- Curl - https://github.com/php-curl-class/php-curl-class (для запросов)
- Nokogiri - https://github.com/olamedia/nokogiri (Парсер)

Все эти библиотеки нужно подключить в проект с помощью Composer.

Все композер пакеты можно найти тут https://packagist.org

Задание которое нужно выполнить :

1 часть:

Нужно спарсить 5 страниц главной хабрахабра.

А именно:

Необходимо собрать 5 страниц самых последних статей на хабре и записать в таблицу в базу данных.

Данные которые нужно собрать:

Название статьи, дата публикации, количество комментариев на момент парсинга.

2 часть:

С помощью Slim PHP написать приложение которое будет выводить информацию о статье

из базы данных.

Пример:

Я открываю страницу site.dev/post/6 , где 6 - это id статьи в базе данных. И мне на экран выводится вся собранная в первой части ТЗ информация о статье.

Тестовое для стажеров

Общая задача :

Сделать систему по учету книг в библиотеке на Laravel . В системе я вижу общий список книг, для быстрой идентификации, книги имеют изображение обложки, на какой полке находятся, или какой читатель библиотеки взял эту книгу и когда ее взял.

Что нужно сделать для выполнения тестового задания:

1. Создать таблицы и модели для следующих сущностей (кто не использует миграции, тот грязный ахтунг) :
- Книга (свойства книги : автор, категория (научное, детективы, история, биография, детское), метки или теги (#пролюбовь, #немцы, #великаяотечественнаявойна), фото обложки, на какой полке хранится, читатель книги.)
- Полка (свойства полки: название полки)
- Метка или тег (свойства : имя метки)
- Категория (свойства : имя категории)
- Читатель (дата регистрации в библиотеке, фио, дата рождения)

2. Создать контроллер и роуты для создания, редактирования, удаления книг, и просмотра списка книг.
3. Написать Unit тесты для созданного контроллера (каждый метод контроллера должен быть описан в тестах.) Использовать встроенный механизм Laravel для тестов.

Контрольные вопросы:

1. Как бы выглядела таблица журнал чтения книги (в которой видно кто и когда брал ее читать, и когда вернет. Вернул ли ее.)
2. Как можно организовать таблицу файлов, если нам нужно будет хранить не только фото обложки книг, но и фото читателей, а так же полок (где примерно находятся)

Задание со звездочкой:

Доделать контроллеры для полок и читателей.

Организовать журнал чтения (видно кто и когда взял книги)

Сделать возможность создавать тег для книги прямо в форме создания\редактирования

В списке пользователей показывать его возраст (сколько полных лет)

В списке книг показывать статус книги (в наличии, или у читателя). Если у читателя, когда вернется.

PHP junior - требования

Фундамент

- Принцип работы HTTP протокола, понимание как работают DNS сервера
- Умение работать в линукс системе, например ubuntu. Уметь работать в командной строке.
- Навыки установки и настройки PHP, Mysql, и Apache. (Необходимо уметь создать новый хост в системе)
- Знание SQL запросов. Понимать различия между MyISAM и InnoDB. Знать как работают LEFT/RIGHT/INNER JOIN’ы.
- Уметь править HTML + CSS
- Знание основ Jquery

PHP

- нужно знать что такое переменные и константы
- типы данных, приведение типов и сравнение данных
- область видимости переменных
- приоритеты операторов
- побитовые операции
- управляющие конструкции
- пространства имён
- Уметь гуглить функции для работы со строками
- Уметь гуглить функции для работы с массивами
- Работа с файлами и файловой системой тоже вполне тривиальная задача
- Изучить как работает подключение файлов
- Понять как происходит работа с сессиями
- Узнать на практике как обрабатывать входные данные с форм
- Поработать с базой данных с помощью PDO

Объектная модель PHP

- наследование

- область видимости свойств и методов
- магические методы
- обработка ошибок с помощью исключений
- абстрактные классы
- интерфейсы
- трейты как замена множественному наследованию
- позднее статическое связывание

Паттерны проектирования

- MVC
- Singleton не всегда вреден
- Factory
- Depedency Injection
- Active Record

Прочее

- Контроль версий, умение пользоватся GIT
- знакомство с каким-нибудь современным фреймворком (Laravel, Slim, Yii)
- Понять и выбрать для себя ORM, знать ее плюсы и минусы
- Умение работать с composer
- виртуализация с Vagrant
- понимание новых фич PHP7
- Twitter Bootstrap как тренд в вёрстке админок
- работа в правильной IDE PHPStorm
- английский на уровне – задать вопрос на stackoverflow и словить минусов

9 учебных проектов для бэкендера

Ранее на Хабре публиковался перевод статьи с Medium'a с подборкой из 8 проектов, которые можно реализовать, изучая новый язык или фреймворк. На мой взгляд, подборка очень неплохая. Проблема в том, что рассчитана она только на фронтендеров и мобильных разработчиков. А я из другого лагеря.

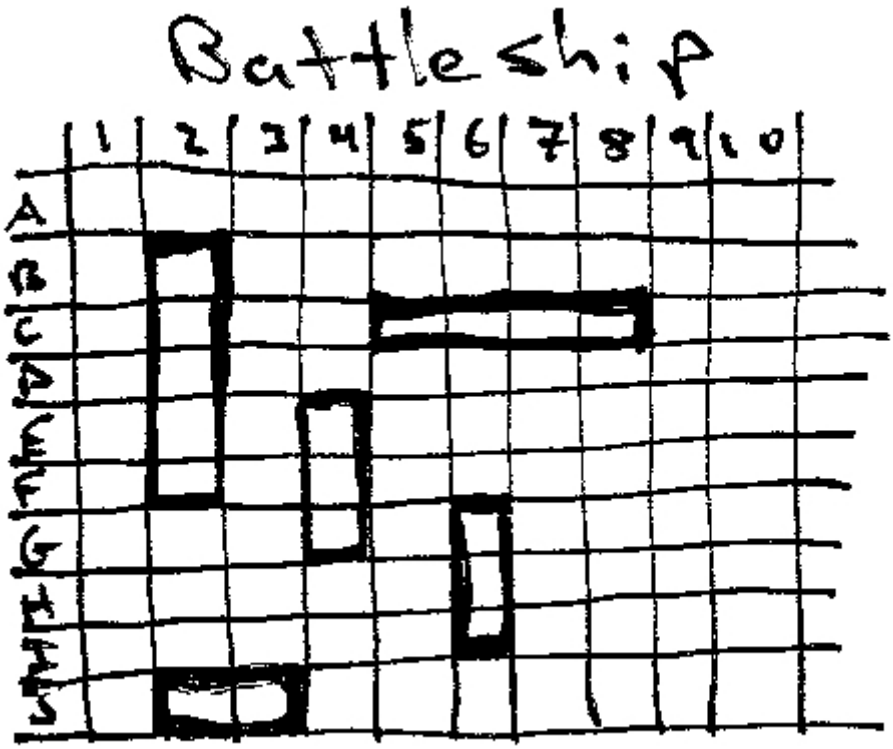
Покопавшись в своих заметках, я собрал для вас подборку идей на тот случай, если чешутся руки сделать что-нибудь своё. Все эти «проекты» уже существуют в том или ином виде, но, иногда интересно написать свой бэкенд-велосипед и проехать на нем по граблям.

1. Движок блога

Да, свой «топ» я решил начать именно с блога. На мой взгляд, это один из простейших вариантов, чтобы изучить тонкости нового ЯП или фреймворка. Посмотрите внимательно на существующие решения и подумайте, что вы бы сделали лучше. Откройте исходники WordPress'a и перечислите, что в нём не так ~~(всё в нем не так...)~~.

Естественно, не стоит пытаться выкатить сразу готовый продукт с кучей фич. Опишите MVP, превратите его в checklist и вперёд! Кстати, этот совет относится и ко всем остальным пунктам.

2. IO-игра



картинка взята отсюда

Если вы хотите вникнуть в реактивное программирование, веб-сокеты и вот это всё, а чат писать слишком скучно, можно попробовать реализовать бэкенд для: крестиков-ноликов 15x15, морского боя или, даже, гомоку. Плюс перечисленных игр в том, что клиент к ним можно создать даже с минимальными знаниями JS.

Бонусом придется подумать над алгоритмом валидации игрового поля, что тоже не так просто, как кажется.

3. Парсер формата файла

Это тот случай, когда вы можете сделать что-то действительно полезное для сообщества вашего любимого языка программирования. Посмотрите, с какими файлами (медиа, документы, данные, и т.д.) вы регулярно работаете и сделайте для них библиотеку на вашем ЯП. Только не пишите обёртку над существующим расширением, а изучите спецификацию файла. И руками, по хардкору.

Как пример приведу библиотеку на PHP для работы с 3d-моделями формата STL. Описание этого формата есть в интернете. Но, еще 4 года назад в сети была только одна рабочая библиотека, которая умела возвращать габариты и объем модели. Ее продавали по 10$ за копию. И она пользовалась большим спросом.

Если же придумать что-то новое не получается, стоит попробовать сделать свою реализацию уже существующего. Например, на сколько мне известно, до сих пор нету ни одной библиотеки на PHP, которая переварила бы >1GB XML-файл, не съев всю ОЗУ. Хотя, есть мнение, что stream piping, SPL и немного упорства должны решить эту задачу.

4. Telegram, Skype, Slack любой другой-бот

картинка взята отсюда

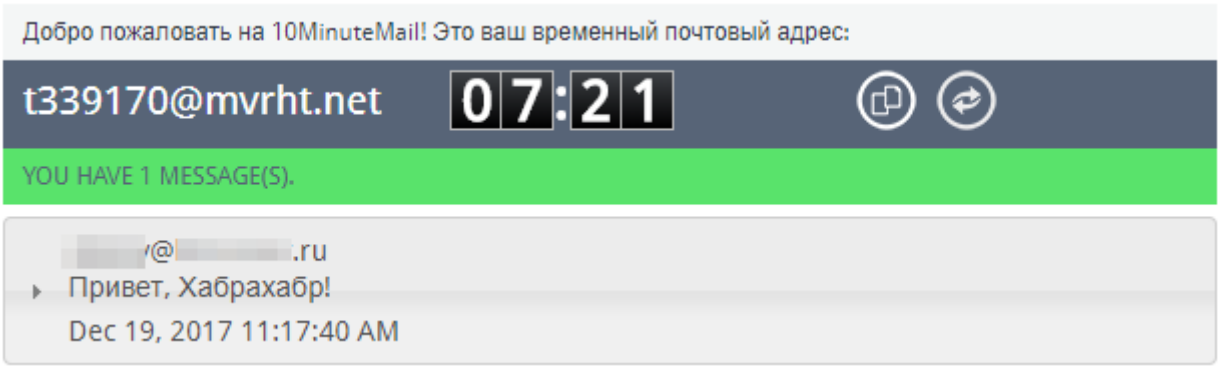
Вариантов тут может быть масса: прогноз погоды, напоминания, калькулятор, карточный «Пьяница», генератор одноразовых паролей для сайта и т.д.

Пусть каждый из них уже кем-то создан. Но, мы ведь с вами хотим научиться чему-то новому и добавить «веса» своему резюме. Не так ли?

5. Движок форума

Плюсы тут те же, что и у первого пункта.
Если вам это кажется банальным и ненужным, почитайте статью о том, как человек написал форум на ассемблере. Зачем? Просто **потому что может**. Зато теперь он может подтвердить свои знания ссылкой на интересную статью и репозиторий к ней. Это ли не мотивация?

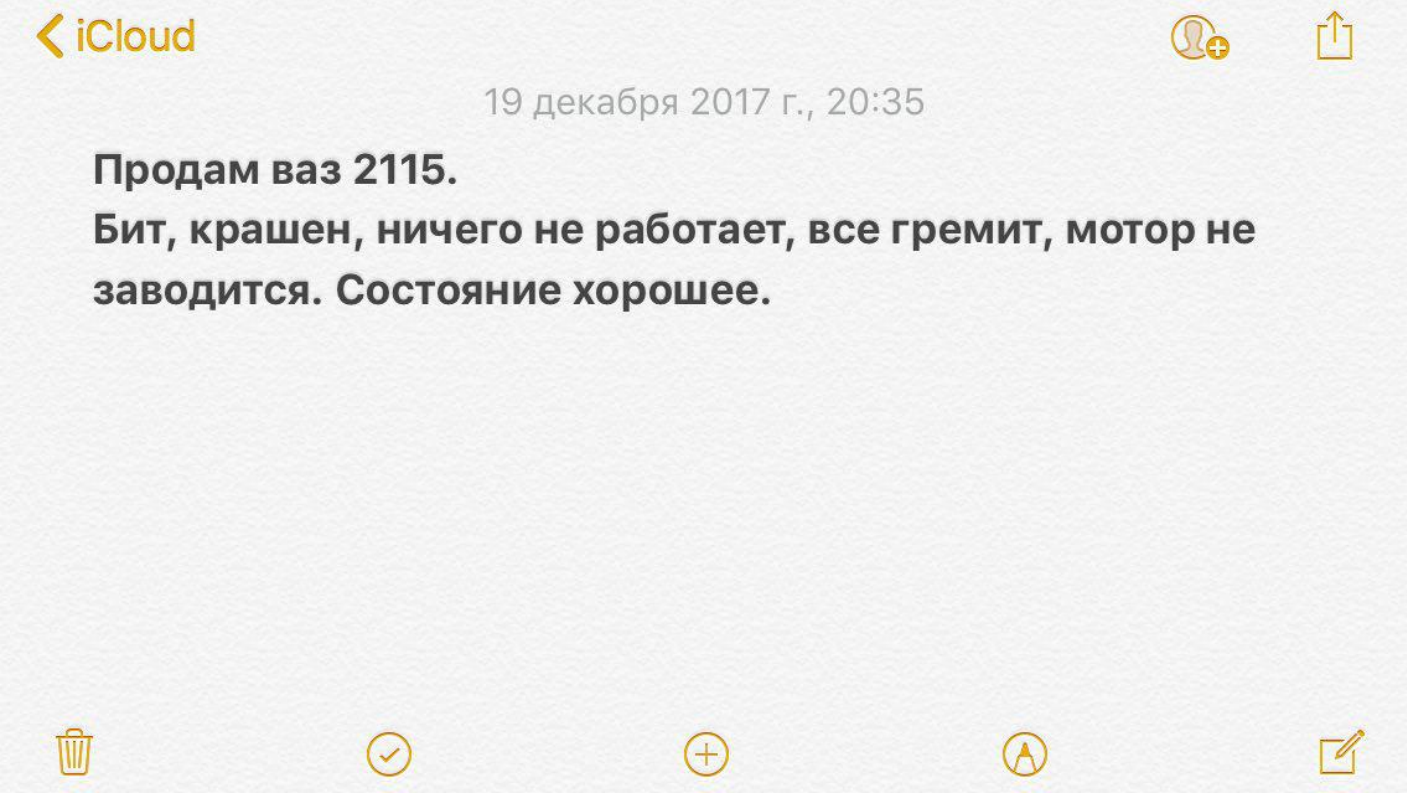
6. Клон 10 Minute Mail



Наверное, всем известен прекрасный сервис 10minutemail. Он позволяет не светить свою почту при регистрации на неизвестных сайтах. Просто получите уникальный email на 10 минут и вводите его везде, не опасаясь навязчивых рассылок.

Если хочется написать что-то небольшое и несложное — это отличный вариант.

7. Генератор изображений с текстом



Очень часто в социальных сетях объявления публикуют в виде скриншота экрана телефона с открытым приложением заметок.

Можно сделать для этих людей сервис для генерации изображений из текста. В гугле я нашел подобное решение, но, на мой взгляд, здесь слишком много лишнего функционала. Достаточно формы для ввода текста и кнопки: «сделать хорошо!».

Кстати, начинающим фронтендерам предлагаю эту задачу для реализации без сервера.

8. Pomodoro

картинка взята отсюда

> Метод «Помидора» — техника управления временем, предложенная Франческо Чирилло в конце 1980-х. Техника предполагает разбиение задач на 25-минутные периоды, называемые «помидоры», сопровождаемые короткими перерывами. — Википедия

Если кратко, то 2 часа работы делятся на 4 отрезка («помидора») по 25 минут, с перерывами в 5 минут. Дабы не засекать все вручную, в интернете есть куча приложений «Pomodoro».

И, я знаю, что этот проект можно создать без сервера. Но я предлагаю сделать это ради фана. И вообще, статья для начинающих бэкендеров, так что, логику пишите на сервере. И прикрутите push-уведомления! И Telegram-бота не забудьте!

9. Пишите свой блог

Нет, не движок блога, а именно технический блог. Один из лучших способов закрепить новые знания — попытаться объяснить их другому. Изучите что-то новое и перескажите это максимально простым и понятным вам языком. Это помогает структурировать в голове

новые знания. К тому же, ваша статья может оказаться для кого-то полезной.

Вместо заключения

Когда-то я сам искал подобный список. Надеюсь, что хоть кого-то он вдохновит на написание чего-то свежего и **своего**. Также, буду рад увидеть продолжение списка в комментариях.

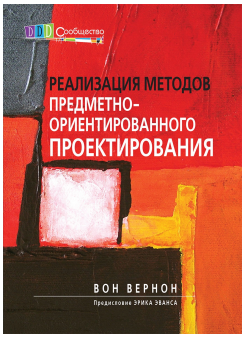
Что почитать

Image not found or type unknown

Более 10 лет первое издание этой книги считалось одним из лучших практических руководств по программированию. Сейчас эта книга полностью обновлена с учетом современных тенденций и технологий и дополнена сотнями новых примеров, иллюстрирующих искусство и науку программирования. Опираясь на академические исследования, с одной стороны, и практический опыт коммерческих разработок ПО - с другой, автор синтезировал из самых эффективных методик и наиболее эффективных принципов ясное прагматичное руководство. Каков бы ни был ваш профессиональный уровень, с какими бы средствами разработками вы ни работали, какова бы ни была сложность вашего проекта, в этой книге вы найдете нужную информацию, она заставит вас размышлять и поможет создать совершенный код.

Книга состоит из 35 глав, предметного указателя и библиографии.

https://www.ozon.ru/context/detail/id/138437220/



"Для разработчиков программного обеспечения любой квалификации, стремящихся повысить свой уровень в области проектирования и реализации предметно-ориентированных промышленных приложений с учетом лучши...

https://www.ozon.ru/context/detail/id/35045716

Программирование — это скучная магия

Есть один карточный трюк, который запомнился мне навсегда. Вот его краткое описание: доброволец выбирает карту и запечатывает её в конверт. Затем фокусник предлагает добровольцу выбрать чай. У него есть десятки коробок чая, и все они упакованы в пластик. Доброволец выбирает одну из коробок, срывает обёртку и выбирает один из упакованных пакетиков с чаем. Потом он вскрывает упаковку, и… внутри оказывается его карта.

Секрет трюка прозаичен, но меня он привёл в восторг. К выбору карты добровольца подталкивают. Однако выбор из этих десятков коробок с чаем на самом деле свободный, и выбор чайного пакетика внутри коробки тоже делается свободно. Здесь нет никакой ловкости рук: фокусник не касается коробок или выбранного добровольцем чайного пакетика. Карта *на самом деле* находится внутри этой упаковки чайного пакетика.

Вся хитрость заключается в подготовке. Перед выполнением фокуса фокусник покупает

десятки коробок чая, вскрывает каждую и разворачивает каждую упаковку с чайным пакетиком. Кладёт в каждую упаковку тройку крестей. Снова запечатывает упаковку. Возвращает упаковки обратно в коробку. Снова запечатывает каждую коробку. И повторяет так сотни раз. На это уходят часы, может быть, даже дни.

«Фокусом» это является именно потому, что такая подготовка выглядит настолько скучной, настолько невозможно монотонной, что когда мы видим трюк, то не можем представить, что кто-то проделал бы столь скучную работу, чтобы добиться такого простого эффекта.

Теллер рассказывает об этом в статье о семи секретах магии:

> “Вас обманет фокус, если в него нужно вложить больше времени, денег и труда, чем захотели бы вложить в него вы (или любой другой здравомыслящий зритель). Мы с моим напарником Пенном однажды достали 500 живых тараканов из цилиндра, стоявшего на столе шоу Дэвида Леттермана. На подготовку этого трюка мы потратили несколько недель. Мы наняли энтомолога, предоставившего нам медленно движущихся тараканов, которые хорошо бы смотрелись на камеру (те, которые живут в наших домах, не будут ждать, пока оператор успеет взять крупный план). Энтомолог научил нас, как доставать насекомых, не вереща при этом, как девятилетние девочки. Затем мы изготовили из пенопласта потайную перегородку (это один из тех немногочисленных материалов, за которые не могут уцепиться тараканы) и придумали хитрую процедуру установки перегородки в шляпу. Вы считаете, что здесь больше возни, чем того стоит фокус? Для вас — возможно, но не для фокусников.

[То самое видео про тараканов с таймкодом, начало трюка на 7:23]

Новички в технической отрасли часто спрашивают меня, в чём секреты успеха в ней. На самом деле их не так много, однако этот секрет — готовность делать нечто столь ужасно монотонное, что это покажется магией — работает и с технологиями.

Наша отрасль одержима автоматизацией, упрощением процессов и эффективностью. В одном из фундаментальных для инженерной культуры текстов Ларри Уолла «Достоинства программиста» упоминается и лень:

> **Лень**: качество, заставляющее вас предпринимать огромные усилия для снижения суммарных затрат энергии. Она заставляет вас писать облегчающие труд программы, которые оказываются полезными для других людей, и документировать написанное вами, чтобы не пришлось отвечать на слишком много вопросов.

Я не могу не согласиться: возможность перекладывать монотонные задачи на программу — одна из лучших особенностей знания кодинга. Однако иногда проблемы нельзя решить автоматизированием. Если ты готов к скучной, однообразной работе, то в глазах других ты будешь выглядеть волшебником.

Например, однажды я пришёл в команду, которая поддерживала утопавшую в багах систему. У них было что-то около двух тысяч открытых баг-репортов. У багов не было никаких меток, категорий и приоритетов. Команда не могла прийти к договорённости о том, какие проблемы нужно устранять первым делом. В результате они, по сути, выбирали баги случайным образом, но было непонятно, важна ли каждая отдельная проблема. Новые баг-репорты невозможно было обрабатывать эффективно, потому что поиск дубликатов оставался практически невозможным. Поэтому количество открытых тикетов продолжало увеличиваться. Команда месяцами буксовала на месте. Передо мной поставили задачу решения этой проблемы: снять команду с мели, обратить тренд количества открытых тикетов, найти способ постепенно снизить его до нуля.

И я использовал тот же трюк, что и фокусник, то есть трюка никакого и не было: я занялся работой. Я распечатал все проблемы — по одной странице на каждую проблему. Я прочитал каждую страницу. Я занял огромный кабинет и начал раскладывать на полу стопки листов. Я писал на стикерах метки и приклеивал их к стопкам. Я перемещал страницы из одной стопки в другую. Я записывал на маркерной доске длинные столбцы номеров тикетов; я представлял себя героем Бена Аффлека из фильма «Расплата». Я провёл в этом кабинете почти три недели, и вышел из него тогда, когда просмотрел, пометил, категоризировал и

приоритезировал каждый баг.

Сразу после этого тренд пошёл в обратную сторону: мы мгновенно смогли закрыть несколько сотен тикетов как дубликаты, а на оценку новых репортов теперь требовались считанные минуты, а не целый день. Если не ошибаюсь, чтобы снизить количество до нуля, понадобился год или больше, но процесс был достаточно спокойным. Люди говорили, что я совершил невозможное, но они ошибались: я просто сделал нечто столь скучное, что никто другой не хотел этим заниматься.

Иногда программирование кажется магией: ты произносишь какое-то таинственное заклинание, и армия роботов исполняет твою волю. Но иногда магия рутинна. Если ты готов и желаешь заниматься скучной работой, то сможешь выполнить невозможное.

оригинал https://habr.com/ru/company/macloud/blog/551838/