Как стать Senior Bitrix Developer?

Дерманов Марк

ARTW

План доклада

- Кто такой senior
- Разница между уровнями
- Как стать специалистом
- Как повысить качество
- Как повысить скорость
- Дальнейшие горизонты

Senior Bitrix Developer

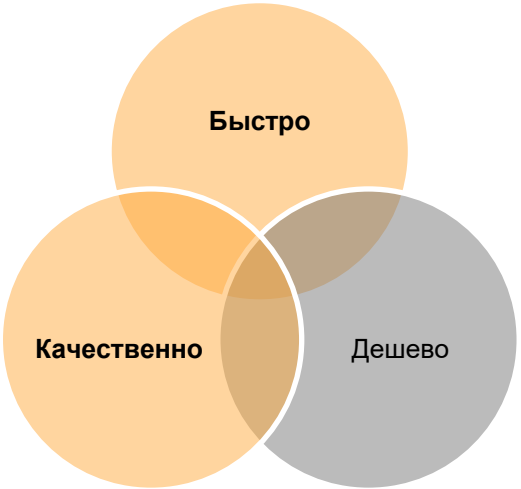
=

Developer(web) * Senior + Bitrix

В чем разница между Junior / Middle / Senior

- Величины относительные
- Где тонкая грань?

Что хочет бизнес



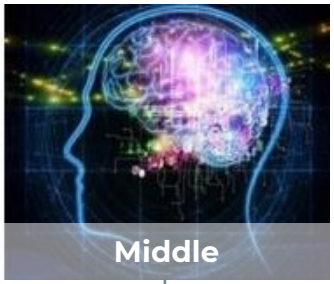
- Быстро – в срок или раньше
- Качественно – без багов
  и чтобы легко поддерживать
- Дешево – выходит за рамки доклада ☺

Технические навыки

Бизнес результат

- Нет опыта
- **Должен уметь программировать**
- Должен знать на базовом уровне html / css / js / php

- Нет результата



- Есть опыт, 1-2-3-4-5 лет
- Знает минимум, чтобы делать работу
- Изучает новое только при необходимости

- Может решать задачи, не всегда самостоятельно
- **Просто делает «эту» работу, «как-нибудь»**
- Долго / некачественно, регулярно есть замечания
- Бизнес часто недоволен

- Есть опыт, 3+ лет
- **Разобрался, как быть эффективным**
- Развивается, знает больше чем нужно

- Самостоятельно решает задачи
- Работает на результат
- **Соблюдает сроки / качество**
- Бизнес доволен

В чем разница между

Middle / Senior

Senior –

самостоятельный специалист,

способный *быстро* и *качественно*

решать задачи бизнеса

- Быстро – сроками доволен бизнес
- Качественно – бизнес доволен проектом, а другие программисты кодом / архитектурой
- Можно убрать любое слово из определения – и получится Middle

Как стать специалистом?

Как стать специалистом

- Нужно ~~стать~~ быть программистом
- Изучить рабочий стек веб-программиста
- Наработать больше 10 000 часов опыта ~ 3-5 лет
- Регулярно развиваться
- Читать, читать, читать КНИГИ!
- «Здесь нужно бежать, чтобы оставаться на месте» (с)

Нужно быть программистом

- Начать обучение с компилируемых языков – c\c++, **c#**, java, pascal
- Научиться писать консольные приложения
- Научиться составлять блок-схемы, UML-диаграммы
- Научиться использовать построчный отладчик
- Изучить и понять алгоритмы сортировок
- Изучить и понять структуры данных
- Изучить и понять основы реляционной алгебры
- Изучить ООП и функциональное программирование

- Подсказка – нужно получить профильное высшее образование

Рабочий стек веб-программиста

- HTML
- CSS
- Javascript

frontend

- PHP
- MySQL
- Bitrix

backend

- Memcache
- Redis

Кеширование

- Apache
- Nginx

Веб-сервер

- ssh
- bash, cron
- php –f, mysql
- htop, mytop

Сервер

- Это не всё, но минимум

PHP – полезные материалы

- php.net
- w3schools.com/php
- Стандарты PSR
- laracasts.com/skills/php
- Литература для веб-разработчика на PHP
- PHP: Правильный путь

MySQL – полезные материалы

- Справочное руководство по MySQL
- w3schools.com/mysql
- Индексы – скорость фильтрации и сортировки
- План исполнения (explain) – когда тормозит
- Книга «MySQL по максимуму»

Bitrix Framework – полезные материалы

- Академия 1С-Битрикс
- Центр поддержки разработчиков
- Сертификация разработчиков - сертификаты
- Сертификация разработчиков – экзамены
- Документация: старое ядро
- Документация: новое ядро
- Сообщество, живая лента
- Форумы
- Tg чат для разработчиков @bitrixfordevelopers
- Vk группа для разработчиков /bitrix_for_developers

Bitrix Framework – уверенный уровень

- Жизненный цикл страницы
- Авторизация / права доступа
- Шаблон сайта
- ЧПУ
- **Комплексный компонент** / Эрмитаж
- Модули / События
- Агенты
- Интеграция с 1С
- ORM / Кеш
- Инфоблоки / HL-инфоблоки
- Каталог / умный фильтр / корзина / заказы
- Старое и новое API
- Знать ядро наизусть ☺
- Не писать в ТП, а разобраться самому

Как повысить скорость?

Как повысить скорость

- Скорость – это соблюдение сроков
- Правильная оценка
- Продуктивный кодинг

Правильная оценка

- Время на поиск информации
- Время на проектирование
- Время на подготовку БД
- Время на описание предметной области
- Время на написание кода ядра
- Время на написание клиентского кода
- Время на ручное тестирование
- Вероятность осложнений
- Небольшой запас
- **Нужно оценивать объем работы**, а не сложность

Продуктивный кодинг

- Локальная копия проекта, рабочая (Open Server, mamp, etc)
- IDE – любая, главное чтобы PhpStorm
- xDebug вместо print_r(), var_dump()

PhpStorm (IDE) – 1/10 всех преимуществ

- static analysis
- code complete
- git / side by side diff
- terminal / ssh
- deploy / sync with remote
- search in project
- go to file / class / function / declaration
- find usages
- bookmarks / hotkeys
- debug
- mysql
- project state / scope
- and many more

Как повысить качество?

Как повысить качество

- Форматирование кода
- Архитектура приложения
- Структура базы данных
- Предупреждение и отлов багов

Как повысить качество

- **Форматирование кода**
- Книга «Как писать совершенный код»
- PHP стандарты оформления кода PSR-1, 2
- Линтеры - PHP-CS-Fixer, PHP_CodeSniffer

Как повысить качество

- **Архитектура приложения**
- Разделить на слои
- Отделить ядро от клиентского кода
- Описать предметную область, классы – контракт
- ООП
- Паттерны проектирования – GRASP, SOLID, GoF
- Автозагрузка классов PSR-4
- Никакой магии
- Статический анализ кода – IDE должна понимать ваш код
- TDD – сначала тесты и клиентский код
- Отделите backend и frontend
- KISS, DRY, YAGNI

Как повысить качество

- **Структура базы данных**
- Сущности и поля
- 3-я нормальная форма
- Типы полей, ограничения (type, size, null)
- Внешние ключи (foreign keys)
- Индексы
- *Написать миграцию*

Миграции

- Всегда есть несколько площадок с проектом
- Изменение структуры БД без админки
- Изменение контента – синхронизация
- Логика в коде привязана к структуре БД
- Проблема: новый код, старая БД – не работает
- Миграция – это php-скрипт
- Создать иблок / группу и тд – через Bitrix API
- Управляет миграциями отдельный модуль
- Скрипты хранятся в системе контроля версий
- Всем понятно, какие были изменения
- Легко обновить БД и автоматизировать

Как повысить качество

- **Предупреждение и отлов багов**
- Строгий режим интерпретатора - error_reporting(E_ALL);
- Логи – соблюдать PSR-3
- Использовать готовый код - composer
- Система контроля версий – git
- Автотесты, регулярный запуск – phpunit, selenium
- Ревью кода – pull request в master
- Парное программирование

Git – система контроля версий

- Книга «Pro Git»
- https://githowto.com/ru
- Git flow
- Заведите аккаунт на Github
- Git bash – начните с консоли
- SourceTree
- GitKraken
- PhpStorm git

Потом GUI клиенты

Итого, чтобы быть Senior

- Нужно быть профессионалом
- Постоянно развиваться
- Нести ответственность
- Работать на результат


Highload
Open source
DevOps
Что дальше?
Другие языки
NoSQL
Фреймворки

Дерманов Марк

Разработчик, руководитель, наставник

mark@dermanov.ru

fb.com/dermanov.ru

artw.ru

THANK U