職務経歴書

基本情報

key	value
氏名	Yuji Miyashita
居住地	東京都
最終学歴	中央大学 理工学部

各種アカウント

- Github
- Qiita
- Wantedly
- YOUTRUST
- note
- Zenn
- SpeakerDeck

言語

Ruby

- menthod_missingのオーバーライドや動的にメソッドを生成するような、Ruby特有のメタプロ構文の読み書きができる
- `each`, `map`, `select`, `any?` あたりのEnumerableモジュールのメソッドの読み書きができる
- pryを使ったデバッグができる

JavaScript/TypeScript

- Chromeや `debugger` を使ったデバッグができる
- 基本的な構文が書ける
- バグ修正や簡単な開発ができる

MySQL

- 基本的な構文が書ける
- クエリからActiveRecordのメソッドを逆引きできる
- EXPLAINでクエリやテーブルの状況を確認できる

フレームワークやライブラリ

Ruby on Rails

- ActiveRecordの `preload`, `eager_load` 等を使ってパフォーマンス改善ができる

- モデル数300-400規模のアプリケーションへの耐性がある
- バリデーションやコールバックが発火する/しないに応じて、ActiveRecordメソッドを的確に使い分けができる
- ビジネスロジックやバリデーションが親クラスに集約されていると、データの流れが一方通行になり、記述がシンプルになることを理解している
- アンチパターンをある程度把握している
- アソシエーション設計ができる
- Railsエンジンの知識がある
- 若手向けの教材を作れる
- メジャーバージョンアップの経験がある

RubyGems

以下のようなGemを扱ってきた

- activerecord-import
- acts_as_list
- acts-as-taggable-on
- apartment
- devise
- kaminari
- nokogiri
- paranoia
- rack
- ransack
- redis
- sidekiq

GraphQL

- フロントとバックの疎通の仕方がわかる
- 基本的な構文が書ける
- graphql-batch を使った開発をしたことがある
- Apollo Client を使った開発をしたことがある

React.js

- 基本的な構文が書ける
- コンポーネント単体で値を保持する `useState` や、コンポーネント間で値を保持する `Provider` を用いてコンポーネントの中身を書ける
- Next.jsに対する知見がある

Backbone.js

- view, template, model, collectionに対する理解がある
- jQueryに依存するフレームワークなので、jQueryも書ける

その他スキル

コマンドライン

- 本番環境にSSHし、vimでのファイル編集や権限の変更ができる
- 基本的なシェルスクリプトを、書いたり実行したりできる

Git

- `Squash` や `Fixup` して適切なコミットにまとめられる
- 開発を進める上で必要なコマンド操作ができる

テスト (Rspec)

- 適切な `context` や `it` の設計ができる
- ネストの深さや共通化のしすぎ回避など、可読性を意識したテストが書ける
- システムに応じたfactory botの設計ができる

命名

- ライブラリや言語のクラス/メソッドと名前被りしない、かつ具体性を持った名前付けができる

問題発見能力

- 障害対応での原因特定までの時間が速い
- 技術的負債になりそうな実装への嗅覚がある

アプリケーション監視 (Newrelic)

- リクエストごとにパフォーマンスを見られる
- 1リクエストのトレースデータを見て、N+1の発生や遅いクエリの特定ができる

自動化

公式ドキュメントを読みながら、基本的な設定ができる。

- Circle CI
- Jenkins
- Github Action

メンバーマネジメント

- 適正の見極めができ、メンバー同士でハレーションの起きづらいチーム作りができる
- 社内の情報収集をし、次に何が起きるかの予測を立てられる
- ハレーションが起きた時、マネージャーにDMですぐに報告ができる
- マネージャーとのコミュニケーションをかなり頻繁にとり、常に認識違いがないように物事を進められる
- (一時的なら) マネージャーも代役として務まる
- 正論だけで攻めず、経緯や状況把握をした上で優先度を決めて改善を進められる

タスクマネジメント

- 適切な難易度のタスクをメンバーにアサインできる
- 難関ポイントへの嗅覚がある
- リスクヘッジができる
- プロジェクトを炎上させないことが得意
- MTGをギリギリまで減らして、メンバーのリソースを最大化できる

- チーム目標やロードマップの策定ができる
- プログラミングスクールのメンター時代に目上の方とのやりとりが圧倒的に多かったので、年齢層関係なくコミュニケーションを取って業務を遂行できる
- 属人化の解消が得意
- 業務フローの整備が得意

コードレビュー

- ぱっと見で例外になるかどうかがわかる
- 障害につながりそうなコードを抽出できる
- システムの仕様に基づいた観点でレビューができる

決済

- 決済の基本的な仕組みがわかる (クレカ, 後払い, アプリ決済)
- システム側でどのような値を保持すれば良いかがわかる
- JavaScriptから決済会社のfunctionを実行し、トークンを受け取った上でサーバサイドに送って、APIを叩く流れがわかる

ドキュメンテーション

- カテゴライズを設計するのが得意
- 何をドキュメントにまとめれば良いかの判断ができる

職務経歴詳細

株式会社SUPERSTUDIO（2017/05〜現在）

販売ページ (LP) の速度改善

数年に渡った機能追加によって販売ページの速度/パフォーマンスが低下していましたので、SQLを見直した上で改善しました。 Newrelicを用いてボトルネックを特定し、コードを修正しました。 負荷試験は以下の2種類を行いました。

- vegetaによる単一リソースへの負荷試験
- AWSのECSを使ってシナリオを組み、世界中のリージョンからリクエストを実行する負荷試験

その結果、速度/パフォーマンス共に改善しました。

- 本番の全ショップ平均速度: 700ms → 300ms
- 無駄なSQLの削減: 184クエリ

管理画面のUI改修

システムの管理画面の新/旧のデザインを切り替えられる状態で、リリースまで完了させました。 期日が短かったので手戻りが許されない中、期日通りにリリースまで完了できたことが良かったです。 携わったメンバーの中で一番システムの仕様を理解していたので、メンバーからの質問対応や技術相談にリソースを当て、メンバー混乱させないように心がけました。

- 期日: 6ヶ月
- プロジェクトに携わった人数: 20人
- 役割: 役員との要件すり合わせ、他部署との連携、周辺の開発、プロジェクト管理
- 改修した画面数: 500

ライブラリのバージョンアップ

RubyやRailsのバージョンアップを行いました。 まずは上げたいRailsのバージョンでは使えないgemを外しました。 具体的には以下の通りです。

- Squeel
- ruby-haml-js
- liquid-rails

これらはシステム全体で使用していたgemでしたので、本体コードへの修正が大幅に入りました。 次にテストコードが揃っていなかったので、開発メンバーを集めて知見共有と、テストコードを強化しました。 上記と並行して開発環境にデプロイし、重要な機能を一通り動作確認しました。 Circle CIにてテストが落ちた箇所の原因調査やコード修正を行い、Railsのバージョンを上げる前後で互換性がある状態を保ちました。 準備を含めて1年半かかりましたが、結果大きな障害を発生させずに、Railsのメジャーバージョンアップを完了させました。

システム全体の保守/運用

本番環境に入って、以下のような作業をしていました。

- クライアントの要望で必要なデータのCSVエクスポート
- CloudWatchを用いたログ調査
- 障害発生時のスクリプト実行
- 欠損データ発生時の本番作業

また様々な障害の復旧作業にて、サービスの品質向上に貢献しました。

基幹システムの開発

- 配送サイクルで「曜日で指定」が選べるように
- ポイントで商品と交換ができる機能
- 顧客結合機能

社内システムの開発

サービスを導入してくださっているクライアントを管理するシステムを開発しました。 各ショップの情報をCronでS3にアップロード、システム側のCronでS3からダウンロード、という構成を取りました。 その結果、権限制御がシンプルになった上、社員による手入力を最小限にできました。 またフロントにTypeScriptやNext.jsの導入をした上で、大幅なリファクタリングも行いました。

- 開発: 2人
- 規模: モデル: 10-20
- 技術: React.js, Rails API, Capistrano, AWS, Next.js, TypeScript

---

株式会社DIVE INTO CODE（2016/02〜2017/08）

大学3年の頃から、インターン生として自分も学びながら受講生にRubyやRailsを教えていました。 受講生は初学者から経験者までさまざまなバックグランドを持った方が集まっていましたので、人によって図解で説明したりコードで伝えたりを使い分けました。 この経験から、専門的な内容を噛み砕いて相手に伝えることが好きになりました。 また受講生サポートだけでなく、DIVE INTO CODE MEDIA のほとんどの記事を執筆したり、Schoo 第6, 7回に出演したりと、様々な角度で貢献しました。