

OBが語る ITエンジニアのリアル

三輪 智樹

LINEヤフー株式会社

今日のゴール

学科・進路選びのヒントを持ち帰る

OBのリアルな経験を共有する

- 大学でどう過ごすと良いかイメージできる
- 進路選択の判断材料が増える
- ITエンジニアという選択肢を知る

まず教えてください

プログラミング経験ある人？

授業でも趣味でも

まず教えてください

エンジニアに興味ある人？

なんとなくでOK

まず教えてください

行きたい学科がすでに決まっている人？

手を挙げてみてください

今日の話は

情報系に進む人以外にも聴いてほしい

学科・進路選びに役立つヒントをお伝えします

今日の流れ

1. 自己紹介

どうやってエンジニアになったのか

2. ITの仕事

エンジニアにも種類がある

3. アプリを設計してみよう

Webエンジニア体験

4. 大学時代の過ごし方

エンジニアになるまで

5. AI時代のエンジニア

これからどうなる？

6. まとめ

今日お話ししたこと

1

自己紹介

どうやってエンジニアになったのか

プロフィール

基本情報

- **出身:** 早稲田実業学校 (2018卒)
- **大学:** 早稲田大学
基幹理工学部 情報理工学科
- **現在:**
LINEヤフー株式会社 エンジニア

やっていること

- 金融の部署でエンジニア
- 全社の生成AI活用Project

高校時代の自分

- 高1: 高等部から入学。陸上部に所属。勉強は普通（テストは平均点くらい）
- 高2: 暗記が苦手だったため理系一択
- 高3: 父の仕事の影響で機械系に興味 → 基幹理工学部 学系2を選択

でも...

当時はプログラミングに興味はなかった...

学部1年での転機

大学1年のC言語授業

- ・何も分からぬ...
- ・友人に課題を助けてもらう
- ・スラスラ解く友人がかっこいい



このままじゃヤバい

- ・長期休みで必死に勉強
- ・少しづつ理解できるように
- ・プログラミングの面白さに気づく

学部2年で情報理工学科へ進学

- 少しうまくいっただけだったが、情報理工学科へ進学を決意
 - 競技プログラミング: アルゴリズムの面白さに気づく
 - アルバイトの誘い: 授業+独学で力をつける
- 入学当初は機械系志望だったが、気づいたら情報系に夢中だった

インターンで実務経験

学部3年～大学院

- ・ インターン・業務委託を開始
- ・ 合計10社で実務経験
- ・ 様々な業界・技術に触れる

気づいたこと

- ・ 授業と実務は全然違う
- ・ 実際に使われるものを作る緊張感
- ・ やってみないとわからない

なぜエンジニアを選んだか

- 1 インターンで実務の面白さを知った
- 2 10社の経験で「これを仕事にしたい」と確信
- 3 学び続ければ成長し続けられる

ポイント：実際にやってみて「合う」と思えた

1章のまとめ

明確な目標がなくても、何があるかわからない

- ✓ 高校時代、エンジニアなんて考えてなかった
- ✓ 大学で出会って、ハマって、進路が変わった
- ✓ 特に学部1年は大切な時期

2

ITの仕事

エンジニアにも種類がある

IT業界の全体像

企画 → 設計 → 開発 → テスト → 運用

エンジニアは「開発」だけじゃない。全部に関わる

グループワーク

「LINE」を作るなら？

どんな役割の人が必要だと思う？

まず1人で考えてみよう

3分間

思いついた役割をメモしてみて

正解はないので、自由に考えてOK！

周りの人と話し合ってみよう

3分間

最低5つの役割を出してみよう

「なぜ必要か」も一緒に考えてみて！

みんなが考えた役割

「何を作るか決める人」「デザインする人」「プログラムを書く人」…

そういう役割、実際に存在します

それぞれの役割に名前がついている。

プロダクトマネージャー (PM)



何を作るか決める人

- ・「既読機能、つける？つけない？」
- ・「スタンプは有料にする？無料にする？」
- ・「次はどの機能を優先して作る？」

チームの方向性を決める、いわば「船長」のような存在

デザイナー



どう見せるか決める人

- ・ トーク画面のレイアウトは？
- ・ 吹き出しの色や形は？
- ・ スタンプの大きさは？

「見た目」だけじゃなく「使いやすさ」も考える

フロントエンド / モバイルエンジニア



画面を作る人

- ・モバイルエンジニア: スマホアプリの画面を作る
- ・フロントエンドエンジニア: Webブラウザの画面を作る

みんながLINEを開いて見ている画面、この人たちが作ってる

バックエンドエンジニア



裏側の処理を作る人

- 送信ボタンを押したらメッセージが届く
- 相手が読んだら「既読」がつく
- 過去のトーク履歴を保存する

目に見えないけど、一番重要な部分かもしれない

インフラエンジニア



土台を作る人

- LINEは日本だけで何千万人も使っている
- 全員が同時にメッセージを送っても落ちない
- データを安全に保存する

サーバーというコンピュータを管理して、サービスの土台を支える

QAエンジニア



品質を守る人

- メッセージが届かない...
- 既読がつかない...
- アプリが突然落ちる...

こういうバグを見つけて報告する。みんなが安心して使えるのはこの人たちのおかげ

実は、これはほんの一部



データエンジニア

PMが判断するためのデータを集めて分析する



機械学習エンジニア

「おすすめスタンプ」などAI機能を作る



プラットフォームエンジニア

他のエンジニアが使う道具や環境を作る

作るものが変われば、必要な人も変わる

ゲームを作るなら？

- 3Dエンジニア
- サウンドデザイナー
- ゲームプランナー

自動運転を作るなら？

- 画像認識エンジニア
- センサーエンジニア
- シミュレーションエンジニア

ITの仕事は本当にいろいろある

2章のまとめ

- 1 1つのサービスを作るには、たくさんの役割が必要
- 2 今日紹介したのは代表的な例。実際はもっと多い
- 3 自分に合った役割が、きっとどこかにある

3

アプリを設計してみよう

Webエンジニア体験

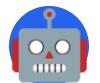
今日このあとやること



LINEみたいなアプリを「作る側」になって考える



最低限必要な機能（MVP）を決める



AIを使った機能を1つ提案する



そのAIのリスクも考える

質問です

LINEのようなメッセンジャーアプリを
0から作るとしたら？

どんな機能が必要でしょうか？

グループワーク①

「最低限必要な機能」を3つ決めてください

- ルール
- 正解はない！理由が説明できればOK
 - 「あつたらいいな」じゃなく「ないと困る」
 - 時間: 3分

周りの人と相談してみよう

グループで共有しよう

5分間

選んだ機能と理由を共有してみよう

- ・自分が選んだ3つの機能を発表
- ・他の人と同じ？違う？
- ・「なるほど」と思ったら取り入れてOK

MVP（最小限の製品）の例

ログイン

本人確認

友達追加

相手を見つける

1:1チャット

メッセージを送る

(通知)

気づける仕組み

これは「正解」ではなく「一例」です

グループワーク②

「最悪の事象」を考えてみてください

このアプリで起きそうな問題は？

時間: 3分

例えば...

なりすまし

他人のふりをする

詐欺リンク

危険なURLをばらまく

乗っ取り

アカウントを奪う

見落とし

重要なメッセージに気づかない

こういった事象は仕様で減らせる



認証（本人確認）

なりすまし・乗っ取り防止



通知機能

重要なメッセージの見落とし防止



送信制限

スパム・ばらまき防止



URL警告

詐欺リンク対策

AIできることを考えてみよう

LINEにAIを組み込むとしたら？

どんな「困りごと」を解決できる？



AIは「困りごとを解決する道具」



便利さだけでなく「リスク」も考えよう

グループワーク③

AIを使った新機能を提案してください

考えること

1. どんな機能？
2. 誰が嬉しい？
3. どんなリスクがある？

時間: 5分

理由も一緒に考えてください

発表タイム

どんなAI機能を
考えた？

リスクもあわせて教えてください

例えばこんな機能はどう？

実際に考えられるAI機能の例

返信提案

文脈に合った返信を提案

詐欺検知

怪しいメッセージを警告

要約・翻訳

長文を要約、外国語を翻訳

タスク抽出

「〇〇して」を自動でリマインド

3章のまとめ

- 1 エンジニアは「何を作るか」を設計する
- 2 便利さだけでなく「守る設計」も大事
- 3 AIは「困りごとを解決する道具」

4

大学時代の過ごし方

エンジニアになるまで

大学6年間でやったこと

- **学部1-2年:** C言語 → 情報系進学・競プロ開始
- **学部3-4年:** インターン → 卒研・業務委託開始
- **修士1-2年:** 研究・就活・業務委託継続 → 内定

学業・研究

授業で学んだこと

✓ C言語

✓ アルゴリズムとデータ構造

✓ ソフトウェア工学

研究で得たもの

✓ 一つのテーマを深掘りする力

✓ 英語で論文を読む・書く力

✓ プrezen・発表の機会

C言語の授業

学部1年の必修科目

課題提出メイン・教場試験なし

当時の状況

- ・授業だけで理解するのは難しかった
- ・分かる人に頼るしかなかった

アドバイス

- ・できる人と仲良くなつておく
- ・少しでも予習しておくと楽
- ・AIに聞くのもあり（概念理解が大事）

C言語の授業で必要な知識

授業で出てくる基本概念

フォルダ構造

ファイルの場所を理解する

条件分岐

if文で処理を分ける

配列・変数

データを入れる箱

ループ

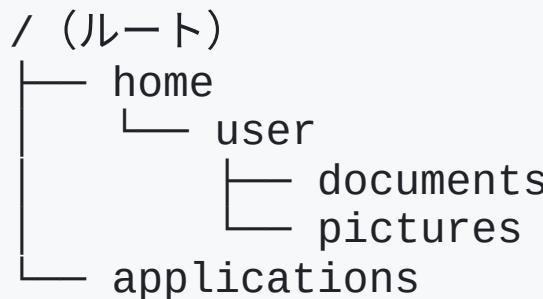
繰り返し処理

ここからクイズ形式で紹介します！

フォルダ構造とは？

パソコンの中身は「階層構造」になっている

イメージ



ポイント

- 「/」から始まる = 一番上
- フォルダの中にフォルダがある
- 「パス」で場所を表す

基本的なコマンド

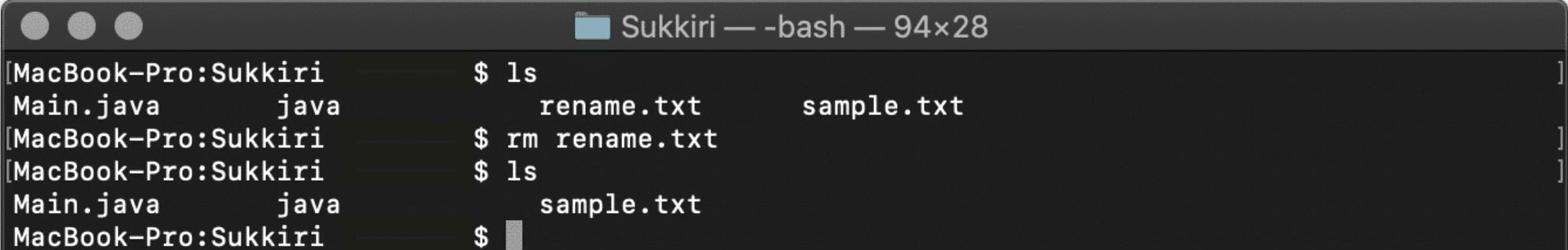
ターミナルで使うコマンド

コマンド	意味	例
cd	フォルダを移動する	cd documents
ls	ファイル一覧を見る	ls
pwd	今いる場所を表示	pwd
..	1つ上のフォルダ	cd ..
.	今いるフォルダ	./program

これらを使ってファイルを操作する

これが使えると…

「黒い画面」が操作できるようになる！



```
[MacBook-Pro:Sukkiri] $ ls
Main.java      java      rename.txt      sample.txt
[MacBook-Pro:Sukkiri] $ rm rename.txt
[MacBook-Pro:Sukkiri] $ ls
Main.java      java      sample.txt
MacBook-Pro:Sukkiri $
```

C言語クイズ①：フォルダ構造

Q. 次のコマンドで移動先はどこ？

現在地: /home/user/documents

コマンド: cd ..

A

/home/user/documents/..

B

/home/user

C

/home

クイズ①：回答

正解：B) /home/user

現在地: /home/user/documents

↓ cd ..

移動先: /home/user

クイズ①：解説

「..」とは？

- ・親ディレクトリを表す
- ・1つ上の階層に戻る
- ・「cd ..」で上に移動

フォルダ構造

```
/home
  └── user           ← ここに移動
      └── documents  ← 現在地
```

覚えておこう：「.」は現在地、「..」は1つ上

次はC言語の問題

コードを読んで
みよう

何が起きるか考えてみて

C言語クイズ②：if文【何が表示される？】

```
int x = 5;  
  
if (x > 10) {  
    printf("A");  
} else if (x > 3) {  
    printf("B");  
} else {  
    printf("C");  
}
```

A

A が表示

B

B が表示

C

C が表示

クイズ②：回答

正解：B

```
int x = 5;
if (x > 10) {           // 5 > 10 ? → ✗
    printf("A");
} else if (x > 3) { // 5 > 3 ? → ✓ ここ !
    printf("B");
} else {
    printf("C");
}
```

クイズ②：解説

1 上から順に評価: 最初に真になった条件だけ実行

2 **else if**: 前の条件が偽のときだけ評価される

3 **else**: どの条件にも当てはまらないとき

ポイント：条件分岐は「上から順番に」チェックされる

C言語クイズ③：配列

Q. 配列の3番目の要素にアクセスするには？

```
int arr[5] = {10, 20, 30, 40, 50};
```

A

arr[3]

B

arr[2]

C

arr[1]

クイズ③：回答

正解：B) arr[2] → 30

```
int arr[5] = {10, 20, 30, 40, 50};  
// [0] [1] [2] [3] [4]  
//  
//  
//  
// 1番目 3番目
```

クイズ③：解説

日常の数え方	プログラミング
1番目、2番目、3番目...	arr[0]、arr[1]、arr[2]...

超重要！ プログラミングでは「0から数える」
これを忘れると「Off-by-oneエラー」というバグの原因に...

C言語クイズ④：整数の割り算

Q. 次のコードでzの値は？

```
int x = 7;  
int y = 2;  
int z = x / y;
```

A

3.5

B

3

C

4

クイズ④：回答

正解：B) 3

```
int x = 7;  
int y = 2;  
int z = x / y; // 7 / 2 = 3 (3.5ではない！)
```

クイズ④：解説

C言語（型を宣言する）

```
int x = 7;      // 整数型
int y = 2;      // 整数型
int z = x / y; // → 3
```

int同士 → 結果もint（切り捨て）

Python（型を宣言しない）

```
x = 7
y = 2
z = x / y # → 3.5
```

自動で小数になる

型とは？ データの種類のこと。int=整数、double=小数など 宣言が必要
C言語はPythonと違い「この変数は整数です」と

C言語クイズ⑤：ループの回数

Q. このループは何回実行される？

```
int count = 0;
for (int i = 0; i <= 5; i++) {
    count++;
}
printf("%d", count);
```

A

5回

B

6回

C

無限ループ

クイズ⑤：回答

正解：B) 6回

```
for (int i = 0; i <= 5; i++) {  
    count++; // i = 0, 1, 2, 3, 4, 5 の6回実行  
}
```

i = 0, 1, 2, 3, 4, 5 → 6回 ループして終了

クイズ⑤：解説

条件	ループ回数	i の値
<code>i < 5</code>	5回	0, 1, 2, 3, 4
<code>i <= 5</code>	6回	0, 1, 2, 3, 4, 5

Off-by-one エラー : 1つずれるバグ

「<」と「 \leq 」の違いで回数が変わる！よくあるミス

覚え方 : 0から始めて「 $< N$ 」なら N 回、「 $\leq N$ 」なら $N+1$ 回

C言語のまとめ

一度理解するとスラスラできる

- 1 最初の壁が一番高い: 知っている人と知らない人で差が出やすい
- 2 英語に似ている: 文法を覚えれば読み書きできるようになる
- 3 一度身につけば一生モノ: プログラミングの土台になる

アルゴリズムとデータ構造

学部2年の科目

数学好きなら絶対ハマる

どんな授業？

- 効率的な問題の解き方を学ぶ
- 理論と実装の両方がある
- 頭の中のイメージを現実に落とし込む

扱う内容

- ソート（並び替え）
- 探索（二分探索など）
- グラフ・木構造

問題：二分探索

1～100の中から数字を当てるゲーム

ルール

- 1～100の中から1つの数字を選ぶ
- 「もっと大きい/小さい」のヒントあり

目標

- 最小の回数で当てたい！

Q. 最悪でも何回で必ず当てられる？

二分探索：回答

答え：最悪7回で必ず当たる！

戦略：毎回「真ん中」を聞く → 候補が半分になる

1回目で 50個 → 2回目で 25個 → ... → 7回目で 1個 に！

二分探索：解説（具体例）

1 「50」 → 「もっと大きい」 → 範囲: 51～100 (50個)

2 「75」 → 「もっと小さい」 → 範囲: 51～74 (24個)

3 「62」 → 「もっと大きい」 → 範囲: 63～74 (12個)

… 繰り返すと → 最悪7回で1個に絞れる！

なぜ7回？ $2^7 = 128 > 100$ なので、7回あれば100個から必ず見つかる

アルゴリズムのまとめ

おすすめ：AtCoder

競技プログラミングで腕を磨こう

1 頭の体操になる：論理的思考力が鍛えられる

2 イメージ→コードの力：考えたことを実装に落とし込む

3 コードを読む力：他人の解法から学べる

4 授業の先取り：基礎ができた状態で臨める

学外での活動：インターン

長期インターンがおすすめ

授業やAtCoderで基礎を固めたら、実務を経験しよう

- 1 実務のノウハウが得られる: チーム開発・コードレビュー
- 2 勉強しながらお金がもらえる: 学びながら収入も得られる
- 3 就活で強みになる: 実務経験は面接で語れる最強の武器
- 4 どこにいても仕事ができる: リモートワークで場所を選ばない

エンジニア就活の特徴

他の職種より差別化が明確



コーディング試験

コードが書ける・書けないがはっきりわかる



成果物が証拠になる

自分の作ったもの・携わった仕事が面接で語れる

大学入学までにできること

- 1 プログラミングに触れてみる: Python、C言語など何でもOK
- 2 PCに慣れておく: タイピング、ショートカットキーなど
- 3 興味のある分野を探す: Web、AI、ゲーム...色々ある
- 4 英語に触れておく: ドキュメントは英語が多い

4章のまとめ

学ぶ → 試す → 実践する

- ✓ C言語・アルゴリズム：最初は難しいが一度身につければ一生モノ
- ✓ AtCoder：論理的思考と実装力を同時に鍛えられる
- ✓ 長期インターン：実務経験は就活の最強の武器
- ✓ 今からできること：プログラミングに触れてPCに慣れる

5

AI時代のエンジニア

これからどうなる？（2026年1月時点）

今日話すこと

いま何が起きている？

エンジニアの価値はどう変わった？

だから、なぜプログラミングを学ぶべき？

まず現実：AIはコードを書ける



AIは「それっぽいコード」を高速に作れる



画面の部品 / 定番の処理 / テスト など



現場でも普通に使われ始めている



だから「勉強いらない？」と思うのは自然

でも重要：AIは"責任"を取れない

AIが出したコードは…

正しい？

仕様に合ってる？

続けられる？

あとから直しやすい？

安全？

危ない穴はない？

✓ 現場で起きていること

「書く」より確認・判断の比重が増えた

価値が下がった能力

(以前ほど重要ではない)



調べてコピペする

AIに聞けば一発で答えが出る



細かいルールを暗記する

覚えなくてもAIが教えてくれる



同じ作業を速くこなす

スピードではAIに勝てない



マニュアル通りに進める

決まった手順はAIが得意



まとめ：「調べる・覚える・繰り返す」はAIに任せる時代

変わらず必要な能力

(AIがいても消えない)



問題設定：何を作るべき？



設計：どう分ける？どう続ける？



デバッグ：なぜ壊れた？



レビュー：正しい？安全？



説明：なぜその判断？



まとめ：エンジニアの仕事＝ 判断の連続

今まで以上に必要な能力



AIを使いこなす力

指示 → 検証 → 修正



品質保証

テスト、監視、再現性



セキュリティ意識



システム全体を見る視点

締めの一言

「AIがあるから勉強しなくていい」ではなく

「AIがあるからこそ、分かる人が強い」

6

まとめ

今日お話ししたこと

振り返り

1 **自分の話:** エンジニアへの道は一つじゃない

2 **ITの仕事:** チームで協力してサービスを作る

3 **設計体験:** 便利さと安全の両立が大事

4 **大学時代:** 今から準備できることがある

5 **AI時代:** 道具として使いこなす側に

最後に伝えたいこと

「進路に迷っても大丈夫。
色々試して、自分に合うものを見つけよう。
ITに興味があれば、ぜひ飛び込んでみて。」

Q&A

なんでも聞いてください！

ありがとうございました！

質問があればいつでも連絡ください

@tamofplease | mmiwatomoki@gmail.com