JVMだけじゃないScala・改

Yuichiro Iwai
2022-03-20
ScalaMatsuri Open Mic Conference

自己紹介

岩井 雄一郎

- アルプ株式会社
- バックエンドエンジニア
- 千葉県在住、二児の父
- Scala/DDD/TDD

本発表の内容は、3/4に開催された「ScalaをSaaSプロダクト開発の裏側お見せします！」の中で発表したLT資料に加筆修正したものです。

Scalaの実行プラットフォームと主要なユースケース

主要なScala実行プラットフォーム

JVM上で実行
Javaの一般的な用途と同等
（バックエンド、GUI、バッチ、etc）

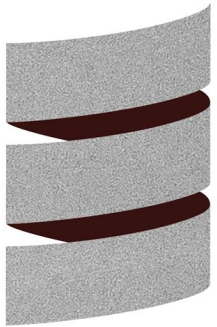
LLVM経由で実行
高速起動を生かしたコマンドラインツール、Cライブラリとの連携など

Scala.js

JVM

Scala Native

JavaScriptとして実行
ブラウザ上、Node.js経由のコマンドライン等



Scala on JVM

- 最も一般的なScalaの利用方法
  - PlayFrameworkやAkkaを用いたバックエンド実装など

みなさま、この用途は詳しいと思いますので、割愛

Scala.js

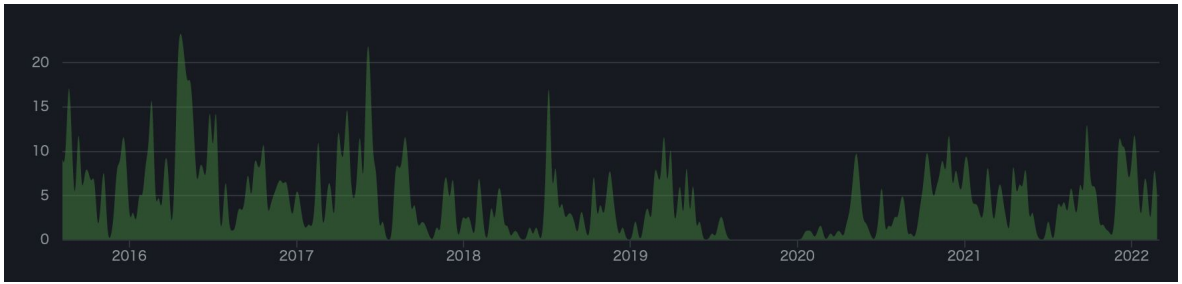
- ScalaのコードをJavaScriptに書き出す、いわゆるAltJS
- Scala3にも対応済み
- 既存のScalaの資産の多くを再利用可能
  - 依存ライブラリがScala.js対応している必要がある
- 既存のJavaScriptの資産と連動可能 (要Facade)

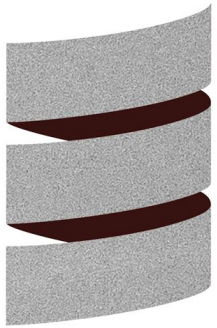
Scala.js

- サーバー/ブラウザ間の強固な連携
  - モデルの共有、通信部分の親和性
- JS(Node.js)経由のソリューションへの展開
  たとえば、AWS Lambdaでの利用など

Scala Native

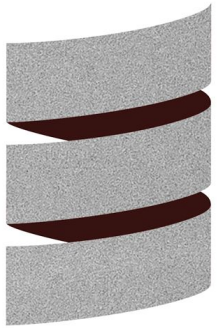
- LLVMを経由してネイティブバイナリにコンパイル
- Scalaの資産に加え、一部のJavaとCライブラリを利用可能
- 最近(v0.4.3, 2022-01-21)Scala3をサポート!
- 一時期、プロジェクトが停滞していましたが、
  Scala Centerのサポートの元、活発になってます！





Scala Native

- 高速な起動を生かしたコマンドラインツール開発
  - 既存のScalaコードを生かした実装が可能
- Cライブラリとの連携の実現
  - Cに対するFacadeを用意すればScala側から呼べる



ライブラリの対応状況

- Scaladex (https://index.scala-lang.org/)で確認できます

主要なFP系のライブラリは概ねJS/Native両方対応
その他でNative対応を進めているライブラリは少なめ...

Scala.jsのメリデメと使いやすいケース

ここからはScala.jsに絞って見ていきます

@yuiwai

投稿日 2019年10月02日　2484 views

Scala.jsを使う意義と使うべきでないケース

Scala，altjs，Scala.js

https://qiita.com/yuiwai/items/f2e302b7375f209e9b2d

Scala.jsを使うメリット

すでにあるScalaの資産を活かせる

- **Scalaで書ける！**
- 既存コードを有効活用して開発出来る
  - ピュアな（ドメイン）ロジックが隔離されている前提条件がある
- Scalaのライブラリが使える

通信(結合)部分の実装コスト削減

- バックエンド/フロントエンドの通信を挟むようなモデルの場合、なんらかの通信規約が必要
- 異なる言語間、異なるプラットフォーム間にまたがると、その管理コストは増大する
  - 型の違い、実装の違い、規約管理コスト、検証コスト...
- バリデーションロジックをフロントに置きたくない（ロジックを分散させたくない）
  - 都度、バックエンドに投げてバリデーションする必要がある

Scala.jsを使うデメリット

プロジェクト構成が複雑化

- 基本的にはクロスプロジェクトの構成をとることになります
  - これにより依存関係が複雑化
    - あるライブラリがJVMでは提供されているけどScala.jsには提供されていない、とか

JVM/Scala.js間で共有するプロジェクトが依存するライブラリを最小限にしておく（ピュアなドメインロジックのみにするのが理想）

ビルド時間が長くなる

- Scalaコードのコンパイル後にJSを書き出す
- Scala.jsの依存解決（Scalaライブラリに加え、JS側ライブラリも）
- 必要に応じてプロジェクト構成の工夫でチューニング

ファイルサイズ

- 最適化（fullOptJS）してもそれなりに大きい
- （コードの規模にもよるものの）一般的なWebページなどで読み込ませるには厳しい
  - 逆に言うとSPA/PWAや、ゲームのようなコンテンツだと、多少の初期読み込みは許容されるので良さそう

FE実装者にScalaを押し付ける危険性

- フロントエンドをScala.jsで作った場合、Scala未経験のフロントエンドエンジニアにScalaで書くことを強要する事態になる危険性
  - Scalaを書いたことがない/書きたくない人が触る可能性のある部分に適用するのは組織的に十分な戦略の検討が必要だと思います
  - JS向けのライブラリだけをScala.jsで提供して、JS/TSから呼び出してもらう、みたいな使い方もアリ

Scala.jsが適したケース

組織内でScala人材が充足している

- Scala書ける ≒ Scala.jsも書ける
    - 手を動かせる人が組織内で確保しやすい

当然ですが、Scalaを書ける人が少ない中で採用すると負債化リスクが高い

内部向け支援ツールの開発など

- この用途でWebツールを開発・運用した経験があります
  - scalajs-reactでフロント書きました
    https://speakerdeck.com/yuiwai/frontend-with-scalajs-react
- 内部向けツールの性質上
  - 細かいディティールよりも、機能が存在することが重視される
    → UXの最適化の優先度が低い
  - コストをあまりかけられない
    → すでにある資産を生かして低コストで実装できた

まとめ

Scala.js使うなら

[must] Scala書ける/書きたい人が組織内に充足している

[should] 再利用可能なScalaコードがある

[may] 内部向けツールなど

Scala.js(やscala-native)という選択肢もあるよ！ということをご記憶いただけたら幸いです

ご清聴ありがとうございました