【MicroProfile開発ガイド】
MicroProfile 概要

2018/12

日本アイ・ビー・エム システムズ・エンジニアリング株式会社

Disclaimer

- この資料は日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム　システムズ・エンジニアリング株式会社の正式なレビューを受けておりません。
- 当資料は、資料内で説明されている製品の仕様を保証するものではありません。
- 資料の内容には正確を期するよう注意しておりますが、この資料の内容は2018年12月現在の情報であり、製品の新しいリリース、PTFなどによって動作、仕様が変わる可能性があるのでご注意下さい。
- 今後国内で提供されるリリース情報は、対応する発表レターなどでご確認ください。
- IBM、IBMロゴおよびibm.comは、世界の多くの国で登録されたInternational Business Machines Corporationの商標です。他の製品名およびサービス名等は、それぞれIBMまたは各社の商標である場合があります。現時点でのIBMの商標リストについては、www.ibm.com/legal/copytrade.shtmlをご覧ください。
- 当資料をコピー等で複製することは、日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム　システムズ・エンジニアリング株式会社の承諾なしではできません。
- 当資料に記載された製品名または会社名はそれぞれの各社の商標または登録商標です。
- JavaおよびすべてのJava関連の商標およびロゴはOracleやその関連会社の米国およびその他の国における商標または登録商標です。
- Microsoft, WindowsおよびWindowsロゴは、Microsoft Corporationの米国およびその他の国における商標です。
- Linuxは、Linus Torvaldsの米国およびその他の国における登録商標です。
- UNIXはThe Open Groupの米国およびその他の国における登録商標です。

目次

本資料について

- 本資料の位置づけ

  本資料は、マイクロサービスをJavaで開発するための標準仕様であるMicroProfileを利用してマイクロサービスを開発する開発者向けに、MicroProfileの各機能の概要およびAPI、サンプルコード等について解説を行うガイドです。対象読者や前提環境は以下の通りです。

  – 対象読者
    - MicroProfileを使用したマイクロサービス開発者
      （WAS Libertyランタイム上でJavaEEアプリ開発経験があることが望ましい ）

  – 前提環境
    - WAS Liberty v18.0.0.3
    - MicroProfile 1.4/2.0

  ※本資料はMicroProfileの以下の機能を対象に記載しております
    - MicroProfile Config
    - MicroProfile Fault Tolerance
    - MicroProfile Open Tracing
    - MicroProfile Health
    - MicroProfile Metrics
    - MicroProfile JWT Propagation
    - MicroProfile Open API
    - MicroProfile Rest Client

マイクロサービス・アーキテクチャーとは

- 以下をカバーするクラウド・ネイティブ・アプリケーション開発・運用のベスト・プラクティス
  - アーキテクチャー
  - 開発・運用チーム・フォーメーション
  - システム・ライフサイクル

- 動機
  - アプリケーション個別の保守を可能にする柔軟なモジュラー構造(サービス)の実現

- マイクロサービス・アーキテクチャー・スタイル
  - 小さなサービスを組み合わせて、 一つのアプリケーションを開発する
  - 各サービスは、 それぞれ独立したプロセスで動作する
  - 各サービスは、 RESTやメッセージングのような軽量な仕組みで通信する
  - 各サービスは、 完全に自動化された仕組みで、 それぞれ個別にデプロイされる
  - サービスは、 それぞれ異なるプログラミング言語で実装できるし、 異なるデータ・ストレージを利用できる

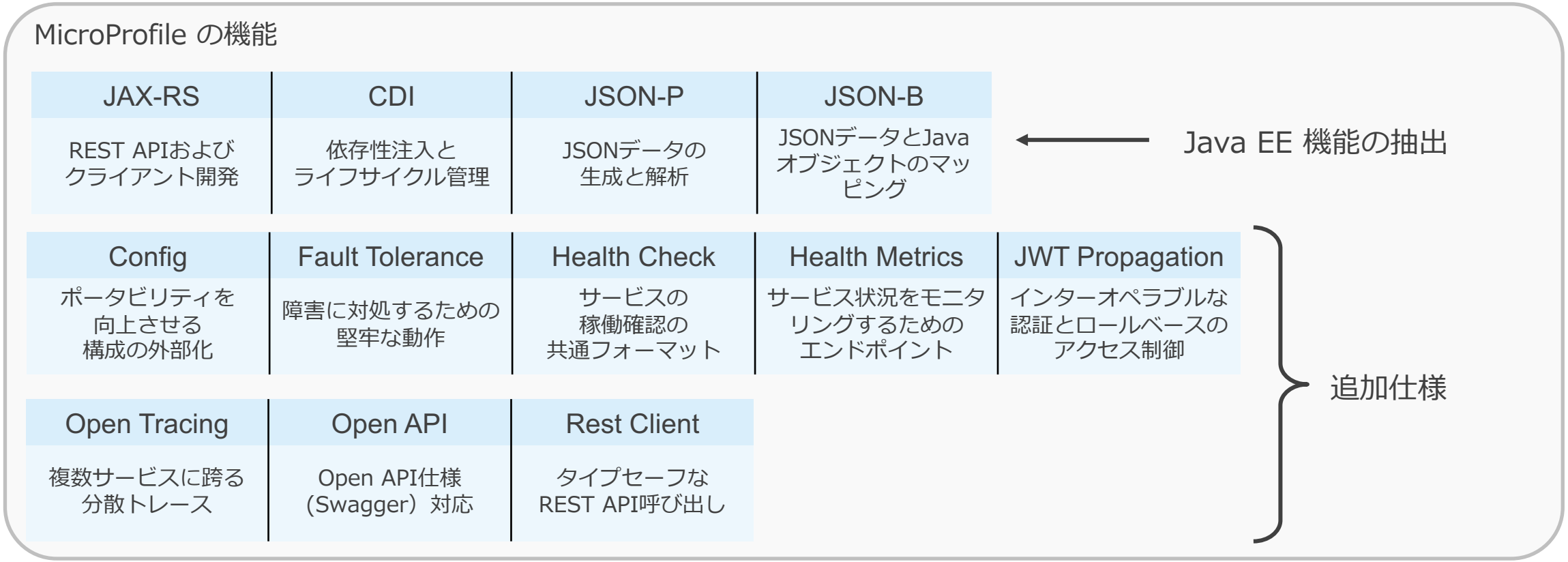
マイクロサービス・アーキテクチャーの特徴

以下のようにクラウド・ネイティブ・アプリケーションの開発・運用をカバーする内容となっています

項目	内容
サービスによるコンポーネント化	ライブラリではなく別プロセスで動作するサービスによってアプリケーションのコンポーネント化を実現している
スマートエンドポイントとシンプルなパイプ	サービス間のメッセージは、HTTP経由でAPI呼び出しされるか、軽量メッセージングシステムによる通信で交換される
障害のための設計	構成されるサービスの障害が起きることを前提に、障害に耐性を持つように設計されている
分散ガバナンス	サービスごとに言語やデータベースなどは統一されず、個別に適切なものが選択される
分散データ管理	サービスごとにデータを持ち、他のサービスがデータにアクセスする場合にはそのデータを管理するサービスのAPIを使用する
ビジネス遂行能力に基づく組織化	役割ごとにチームが構成されるのではなく、複数の役割が混在したチームが各サービスを開発運用する（コンウェイの法則）
プロジェクトではなくプロダクト	サービスは期限のあるプロジェクトとして開発されるのではなく、継続的なプロダクトとして提供される
インフラ自動化	サービスのビルドやテスト、デプロイなどの自動化により継続的デリバリーが実現されている
進化的設計	各サービスごとに変更が行なわれ、漸進的に設計がされる

MicroProfileとは

MicroProfile は、エンタープライズ Java をマイクロサービスの実装用に最適化することを目的としたコミュニティーです。エンタープライズ Java として Java EE が存在しますが、マイクロサービスの開発で必要となるのは、Java EE が提供する多種多様な機能の一部だけです。MicroProfile では、マイクロサービスに必要な Java EE の機能を抽出し、さらに追加の機能仕様を規定することで、マイクロサービスの実装に最適なエンタープライズ Java の仕様策定を目指しています。



LibertyのMicroProfile フィーチャー一覧

WAS LibertyではMicroProfileをサポートしており、以下のフィーチャーを有効にすることにより MicroProfileの各機能を利用することが可能です。

- v17.0.0.3よりMicroProfile 1.2をサポートし、v18.0.0.3(2018年9月リリース)よりMicroProfile 1.4/2.0をサポート

  (WAS Libertyは、v8.5.5.6でJava EE 7を、v18.0.0.2でJava EE 8をサポート)

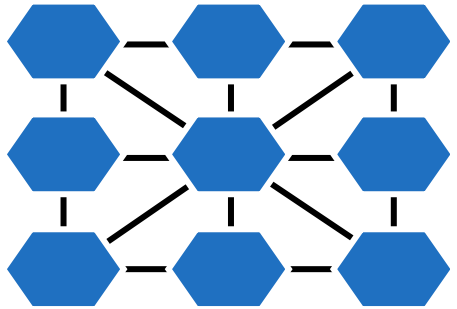
名称	バージョン	Libertyフィーチャー
MicroProfie（全体）	1.4 / 2.0	microProfile-1.4 / microProfile-2.0
Config	1.3	mpConfig-1.3
Fault Tolerance	1.1	mpFaultTolerance-1.1
JWT Propagation	1.1	mpJwt-1.1
Open Tracing	1.1	mpOpenTracing-1.1
Rest Client	1.1	mpRestClient-1.1
Metrics	1.1	mpMetrics-1.1
Health Check	1.0	mpHealth-1.0
Open API	1.0	mpOpenAPI-1.0

（参考）サービス・メッシュとは

サービス・メッシュとは、マイクロサービス共通のプラットフォーム・レイヤーを設けるというMicroProfileとは異なるアプローチでマイクロサービスのための各種機能を透過的に提供するレイヤーです。

- サーキット・ブレーカーなど一部MicroProfileと同様の機能を提供します
- サービス・メッシュの実装としてIstioが現在注目されています

- 主にマイクロサービス間のオーケストレーション機能を提供
  - トラフィックに対する動的なロードバランス
  - ルーティングルールによる適切な粒度のトラフィックコントロール
  - サービス間トラフィックの暗号化や認証
  - サービス全体に対するポリシーの管理と適用
  - 詳細レベルのモニタリングとトレース



- Istio – サービス・メッシュを実現するためのS/W （https://istio.io/）
  - Google、IBM、Lyftが共同で始めたオープンソース・プロジェクト
  - Sidecarモデルによりサービスのアプリケーション・コードを変更することなくサービス・メッシュに必要な機能を提供することが可能
  - MicroProfileと異なりサービスの開発言語は問わない

End of File