既存アプリケーションを Java11に対応させる際に 知っておくべきこと

ikeyat

@2018/11

自己紹介

- 日本Springユーザ会(JSUG)スタッフ
- 某SIerでソフトウェアアーキテクト

  （実態は管理業務多め・・・）

- 某書の筆者の一人

  https://www.shoeisha.co.jp/book/detail/9784798142470

おことわり

- Javaのライフサイクルの話はしません
- Javaの各社サポート等の話はしません

LINE Fukuoka　きしださんの記事

https://qiita.com/nowokay/items/edb5c5df4dbfc4a99ffb

NTT 久保田さんのスライド

https://www.slideshare.net/YujiKubota/introduction-to-java-11-support-and-jvm-features

@u-tanickさんの記事

https://qiita.com/u-tanick/items/bb166929a58a4c20bb88

- 最後までベストプラクティスは出ません

モチベーション

- Javaの今後やサポートについて不安が広がる。仕方のないこと。
- ただ、中身も知らず触ったこともないものをとやかく言う風潮は嫌だ。
- 賛否は抜きにして、少しでも多くの人が新しいJavaに触れたり考えたりするきっかけを作りたい。
- 「●●のことは嫌いでも、Java11のことは嫌いにならないでください！」

Agenda

- JavaのVer追従の必要性
- JavaのVer追従の課題
- Java8⇔Java11の非互換概要
- JavaライブラリやSpringのJava11サポート状況
- Spring4アプリケーションをJava8->11に対応させてみる（非推奨）
- まとめ

JavaのVer追従の必要性

- Java8以前から
  - セキュリティ等パッチ提供
  - 関連製品のサポート継続性
    - 新機能の活用、性能向上
- Java9以降での考慮
  - Javaのライフサイクルが変わり変化が早くなった
  - 無償で利用できるLTS製品が不透明

JavaのVer追従の必要性

- Java8以前から
  - セキュリティ等パッチ提供
  - 関連製品のサポート継続性
  - 新機能の活用、性能向上
- Java9以降での考慮
  - Javaのライフサイクルが変わり変化が早くなった
  - 無償で利用できるLTS製品が不透明

軽くおさらい

大規模エンタープライズ系
システム関係者のあるある

とても重要で敏感！
サポートという言葉さえもらえれば他は何もいらない

あまり興味ない
調査・検討だるい

Java保守ベンダを呼び出す
脱Javaを検討しだす

Java9 新機能

https://codezine.jp/article/detail/10459
https://www.slideshare.net/YujiKubota/java9-69782018

- モジュールシステム(Jigsaw、Javaコアのモジュール分割)
- JShell
- Reactive Stream(Interfaceのみ提供)
- Compact String
- コレクションファクトリの追加
- JavaDocの改善

Java10 新機能

https://codezine.jp/article/detail/10872
https://qiita.com/nowokay/items/d9bc4b3f715d17c2830d

- ローカル変数時のvarを用いた型推論
- Optionalクラス・コレクションクラスなどの API変更
- ガベージコレクションに対する改善とインターフェースの提供
- アプリケーションクラスも含めたクラスデータ共有(CDS)
- ルート証明書の提供（OracleJDKは従来から）
- Docker環境下での改善
- 実験的なJavaベースのJITコンパイラ(Graal)

Java11 新機能

https://codezine.jp/article/detail/11071

- ラムダ式でのvarを用いた型推論
- 新しいHTTP Client（HTTP2/WebSocket）
- シングルJavaファイルからの即時Java実行(コンパイル不要)
- Unicode 10のサポート
- 新しいガベージコレクション方式の追加(ZGC)
- セキュリティ機能・APIの強化（TLS1.3のサポート・暗号・署名機能の強化）
- Javaのモニタリング機能の強化（Flight Recorderの追加等）

Agenda

- JavaのVer追従の必要性
- JavaのVer追従の課題
- Java8⇔Java11の非互換概要
- JavaライブラリやSpringのJava11サポート状況
- Spring4アプリケーションをJava8->11に対応させてみる（非推奨）
- まとめ

JavaのVer追従の課題

- 非互換がゼロではない＆文書化されていない場合
  - 例：Java8でのString.split()
- 利用している製品やライブラリが追従をしない場合
  - 例：Spring2系やdjUnitは、Java8で動作困難
- Java9以降での考慮
  - スリム化のための機能削減が活発化、対象の置換先
  - Deprecated→Removeのペースが高速化

JavaのVer追従の課題

- 非互換がゼロではない＆文書化されていない場合
  - 例
- 利用
  - 例：Spring2系やdjUnitは、Java8で動作困難
- Java9以降での考慮
  - ス
  - D

非互換影響調査、実機検証orテストが必要
→CI/CDしていないと多大なコスト

場合

置換先

素早く変化を察知し将来に備える必要がある
→CI/CDしていないと時間軸に追いつけない

Agenda

- JavaのVer追従の必要性
- JavaのVer追従の課題
- Java8⇔Java11の非互換概要
- JavaライブラリやSpringのJava11サポート状況
- Spring4アプリケーションをJava8->11に対応せてみる（非推奨）
- まとめ

Java9 主な非互換

- JavaのVersionスキーム変更に伴うjava.versionプロパティ値のフォーマット変更（1.8.0_133 -> 9）
- JAXB/JAX-WS等の組込Java EEモジュールがデフォルトで解決されない（-add-moduleで有効化可能）
- JDK内部API利用時のWarning出力
- sun.misc.BASE64Encoder/Decoderの削除（java.util.Base64に移行）
- com.sun.image.codec.jpegの削除（Java Image I/O APIに移行）
- 変数名に「_」が使用できなくなる
- Arrays.asList().toArrays()の戻り値が配列cloneからObject[]に変更
- Stringの内部データがchar[]からbyte[]に変更

Java9 主な非互換

- デフォルトのガベージ・コレクタがG1に（従来はParallel GC）
  ＋GCログ設定オプション変更
- hprof廃止、jhat廃止、VisualVM廃止
- Java DB (旧Apache Derby)廃止
- Native2ascii廃止＋デフォルトでUTF-8でプロパティファイル読み込みに変更
- jigsawに伴うJDK内部ディレクトリ構造の変化、rt.jar/tools.jarの廃止

https://docs.oracle.com/javase/jp/9/migrate/toc.htm
https://builder.japan.zdnet.com/sp_oracle/35095997/
https://www.slideshare.net/YujiKubota/java9-69782018

Java10 主な非互換

- Swingの古いLookAndFeelの削除
- Doclet(javadoc) 旧APIの削除(Java9で刷新のため)
- javahの削除(javac -hで代替)
- -d32/-d64の廃止
- その他マイナーなAPIの廃止

https://docs.oracle.com/javase/jp/10/migrate/toc.htm

Java11 主な非互換

- JAXB/JAX-WS等の組込Java EEモジュールが完全削除(Tomcat利用時注意)
  - JAXB/JAX-WS/JTA/Annotation/CORBA
- sun.misc.Unsafe.defineClassの削除(AOP/Mock系注意※後述)
- Java Script Engine (Nashorn)を非推奨レベルに変更
- Pack200形式の圧縮ツールとAPIを非推奨レベルに変更

Java11 主な非互換

OracleJDK独自の非互換

- JavaFX関連モジュールの削除
- Applet/Java Web Start削除
- JRE廃止、コンパネ廃止
- Mission Controlの同梱取りやめ
- 一部のルートCAの削除

https://docs.oracle.com/en/java/javase/11/migrate/index.html

https://www.oracle.com/technetwork/java/javase/11-relnote-issues-5012449.html#JDK-8193033

Agenda

- JavaのVer追従の必要性
- JavaのVer追従の課題
- Java8⇔Java11の非互換概要
- JavaライブラリやSpringのJava11サポート状況
- Spring4アプリケーションをJava8->11に対応させてみる（非推奨）
- まとめ

非互換影響が大きいライブラリ

- Java11のsun.misc.Unsafe.defineClassの削除の非互換影響を受けるライブラリが多い
- バイトコード操作系
  - ASM：代替機能に対応済み(7.0+)

    https://asm.ow2.io/versions.html
- CGLIB：ASM7.0導入で対応済み(3.2.9+)

  https://github.com/cglib/cglib/commit/fd9f5021c94250baf8f2701576cc8db1a500908e
- Byte buddy：ASM7.0導入で対応済み(1.9.0)

  https://github.com/raphw/byte-buddy/blob/master/release-notes.md
- Javassist：代替手段へ切り替え対応された(3.24+)

  https://github.com/jboss-javassist/javassist/issues/194

非互換影響が大きいライブラリ

- AOP系
  - AspectJ：対応済み(1.9.2+)

    https://www.eclipse.org/aspectj/doc/released/README-192.html

  - Spring AOP：
    - 従前よりCGLIBやASMをリパッケージしている
    - 5.1.0で対応したが、5.1.1でBug Fix(ASM7.0/CGLIB3.2.9相当)

      https://jira.spring.io/browse/SPR-17371

非互換影響が大きいライブラリ

- Mock系
  - Mockito：private/finalクラスのモック化のみ未対応

    https://github.com/mockito/mockito/issues/1483

  - PowerMock：未対応？

    https://github.com/powermock/powermock/issues/904

- EasyMock：4.0.1で対応済み

  https://github.com/easymock/easymock/releases/tag/easymock-4.0.1

非互換影響が大きいライブラリ

- その他
  - JaCoCo：対応済み(0.8.2)

    https://github.com/jacoco/jacoco/issues/663

  - Lombok：対応済み(1.8.14+)

    https://github.com/rzwitserloot/lombok/commit/182cb0cb9e8db6341fb4633c3849b5e90ba6d088

SpringのJavaサポート

- **Spring5.1以上でないとJava11サポートしない**※
  と提供元が明言（2018/11時点）
- Spring5.1にVerUPしてJava11対応するのが推奨される

Please upgrade to Spring Framework 5.1 (and the corresponding Spring Boot 2.1) for JDK 11 support, as the common Long-Term Support migration path from JDK 8. No earlier Spring versions are officially supported on JDK 11, in particular not with JDK 11 bytecode level. Note that third-party components might not fully support JDK 11 yet, so you are likely to be limited in your full-stack options

引用：

https://github.com/spring-projects/spring-framework/wiki/Spring-Framework-Versions#jdk-version-range

※wiki以外にもPivotal講演等での資料投影による説明あり

SpringのJavaサポート

（2018/9 時点 Spring発表）

	リリース予定	Java8対応	Java9対応	Java10対応	Java11対応	Java12対応	EOL予定
Spring4.3	2016/6	x					2020/6
Spring5.0	2017/9	x	x				2019/3
Spring5.1	2018/9	x	x	x	x		2019/12
Spring5.2	2019/6	x	x	x	x	x	未定

Javaライフサイクル変更発表前は、2019/12とも言われていたが延期した模様

情報源（SpringOne Platform 2018 Keynote）：

https://youtu.be/onZJ8beVEtI?t=628

Spring5.1にVerUP?

- Spring4 -> 5.1のVerUPでの非互換は？
  - 一部の外部ライブラリ連携機能削除を除き、Spring自体の大きな非互換はない※（未文書化の非互換はあるかも）
    - 多数の外部依存OSSライブラリが軒並みVerUPされており非互換が見切れない→事実上CI/CD無しだと見切れない
- Spring5.1のEOLは？
  - Spring4.3より先にEOL→一過性のVerUPでは一時凌ぎ

※参考　https://github.com/spring-projects/spring-framework/wiki/Upgrading-to-Spring-Framework-5.x

Spring5.1にVerUP?

- S
  - 大規模エンタープライズ系システム関係者への悪魔の囁き
  - 今使ってるSpring4をJava11で動かせないのか？
- Spring5.1のEOLは？
  - Spring4.3より先にEOL→一過性のVerUPでは一時凌ぎ

※参考　https://github.com/spring-projects/spring-framework/wiki/Upgrading-to-Spring-Framework-5.x

Agenda

- JavaのVer追従の必要性
- JavaのVer追従の課題
- Java8⇔Java11の非互換概要
- JavaライブラリやSpringのJava11サポート状況
- Spring4アプリケーションをJava8->11に対応させてみる（非推奨）
- まとめ

Spring4はJava11で使えるか？

- 公式サポートは諦め、Spring4をJava11で使い続けることはできるのか？（悪魔の果実）

→Spring4がJava11でどの辺りで問題が起きそうか、あたりを付けたい

→Spring4のJUnitをJava11で動かせば、テストケースFailで洗い出せるのでは？

Spring4をJava11でTest試すも・・・

```
$ java -version
java version "11" 2018-09-25
Java(TM) SE Runtime Environment 18.9 (build 11+28)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11+28, mixed mode)

$ git clone https://github.com/spring-projects/spring-framework.git
$ cd spring-framework
$ ./gradlew test                    # masterブランチはSpring 5.1+のため当然Success
(中略)
BUILD SUCCESSFUL in 20m 10s
185 actionable tasks: 185 executed

$ git checkout -b 4.3.x remote/origin/4.3.x
$ ./gradlew test                    # Spring4.3でテスト実行してみるも・・・
FAILURE: Build failed with an exception.

* What went wrong:
Could not determine java version from '11'.
```

Spring4が利用するビルドツールGradleのバージョンが古くてJava11で全く動かず挫折。時間があればGradleのVerUPから始まる遠い旅へ。

※Gradleは4.9+でJava11対応　https://github.com/gradle/gradle/issues/5120

Spring4はJava11で使えるか？（again）

- 公式サポートは諦め、Spring4をJava11で使い続けることはできるのか？（悪魔の果実）

→Spring4がJava11でどの辺りで問題が起きそうか、あたりを付けたい

→Spring4で作られたアプリを実際に動かしてみて、どのような影響があるか確認してみる

Spring4アプリを動かしてみる

- TERASOLUNA Framework ver.5が公開するサンプルアプリ(tourreservation)のSpring4.3版を使用
  - Spring 4.3 & MyBatis 3.4 @ Java8
  - JUnitテストや動作確認用Seleniumテスト付き

    https://github.com/terasolunaorg/terasoluna-tourreservation-mybatis3/tree/5.4.x

- OracleJDK11を無償範囲目的※で利用

  ※ https://www.oracle.com/technetwork/java/javase/terms/license/javase-license.html

Spring4アプリを動かしてみる

Spring4アプリを動かしてみる

1. 環境用意（JDK11/アプリ資材）
2. とりあえずビルド&JUnit
3. selenium走行 /w Tomcat9
4. 目視で動作確認

※１～３だけならTravisCI上でも実施可能なので、ローカルに環境作らずに試すことも可能

エラー遭遇その１

- Maven Buildで謎エラー１

```
$ mvn install
(中略)

[INFO] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 32.529 s
[INFO] Finished at: 2018-10-09T18:06:26Z
[INFO] Final Memory: 23M/77M
[INFO] ------------------------------------------------------------------------
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-source-plugin:2.4:jar
(source-jar) on project terasoluna-tourreservation-env: Execution source-jar of goal
org.apache.maven.plugins:maven-source-plugin:2.4:jar failed: An API incompatibility was
encountered while executing org.apache.maven.plugins:maven-source-plugin:2.4:jar:
java.lang.ExceptionInInitializerError: null
(中略)
```

原因詳細は不明だが、maven-source-pluginが怪しいので、pluginをMavenデフォルトから最新VerUPしてみる

2.4 -> 3.0.1

エラー遭遇その２

- Maven Buildで謎エラー２

```
$ mvn install
(中略)

[INFO] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 01:01 min
[INFO] Finished at: 2018-10-10T17:10:23Z
[INFO] Final Memory: 46M/160M
[INFO] ------------------------------------------------------------------------
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-war-plugin:2.5:war
(default-war) on project terasoluna-tourreservation-web: Execution default-war of goal
org.apache.maven.plugins:maven-war-plugin:2.5:war failed: An API incompatibility was
encountered while executing org.apache.maven.plugins:maven-war-plugin:2.5:war:
java.lang.ExceptionInInitializerError: null
(中略)
```

原因詳細は不明だが、maven-war-pluginが怪しいので以下略

2.5 -> 3.0.0

Spring4.3

エラー遭遇その３

- Maven Buildで謎エラー３

```
$ mvn install
(中略)

[INFO] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 7.091 s
[INFO] Finished at: 2018-11-14T03:23:55+09:00
[INFO] Final Memory: 34M/120M
[INFO] ------------------------------------------------------------------------
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:
3.1:compile (default-compile) on project terasoluna-tourreservation-domain: Fatal error
compiling: java.lang.ExceptionInInitializerError: com.sun.tools.javac.code.TypeTags -> [Help
1]

(中略)
```

検索するとどうやらLombokの問題の模様。
LombokをVerUPする。
1.16.18 ->1.18.2

Spring4.3

エラー遭遇その４

- JavaEEクラス削除によるコンパイルエラー

```
$ mvn install
(中略)

[INFO] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 6.428 s
[INFO] Finished at: 2018-11-14T03:33:4 5+09:00
[INFO] Final Memory: 36M/127M
[INFO] ------------------------------------------------------------------------
[ERROR]/(中略)PriceCalculateSharedServiceImpl.java:[67,6] シンボルを見つけられません
[ERROR] シンボル:   クラス PostConstruct
[ERROR] 場所: クラス
org.terasoluna.tourreservation.domain.service.tourinfo.PriceCalculateSharedServiceImpl
(中略)
```

SpringDIコンテナ初期化時に呼び出す目印として
JavaEEのアノテーションが使われている。この
JavaEEアノテーションはJAX-WSと共にJava11で
削除されたため

Spring4.3

エラー遭遇その４

- JavaEEアノテーションの置き換えライブラリの追加例

pom.xml

```
<dependency>
    <groupId>javax.annotation</groupId>
    <artifactId>javax.annotation-api</artifactId>
    <version>1.3.1</version>
</dependency>
```

JDKに内蔵されなくなったため、
外部から調達するようにする
(旧JavaEE、現JakartaEE)

バージョンは一旦仮で選ぶ

Spring4.3

エラー遭遇その５

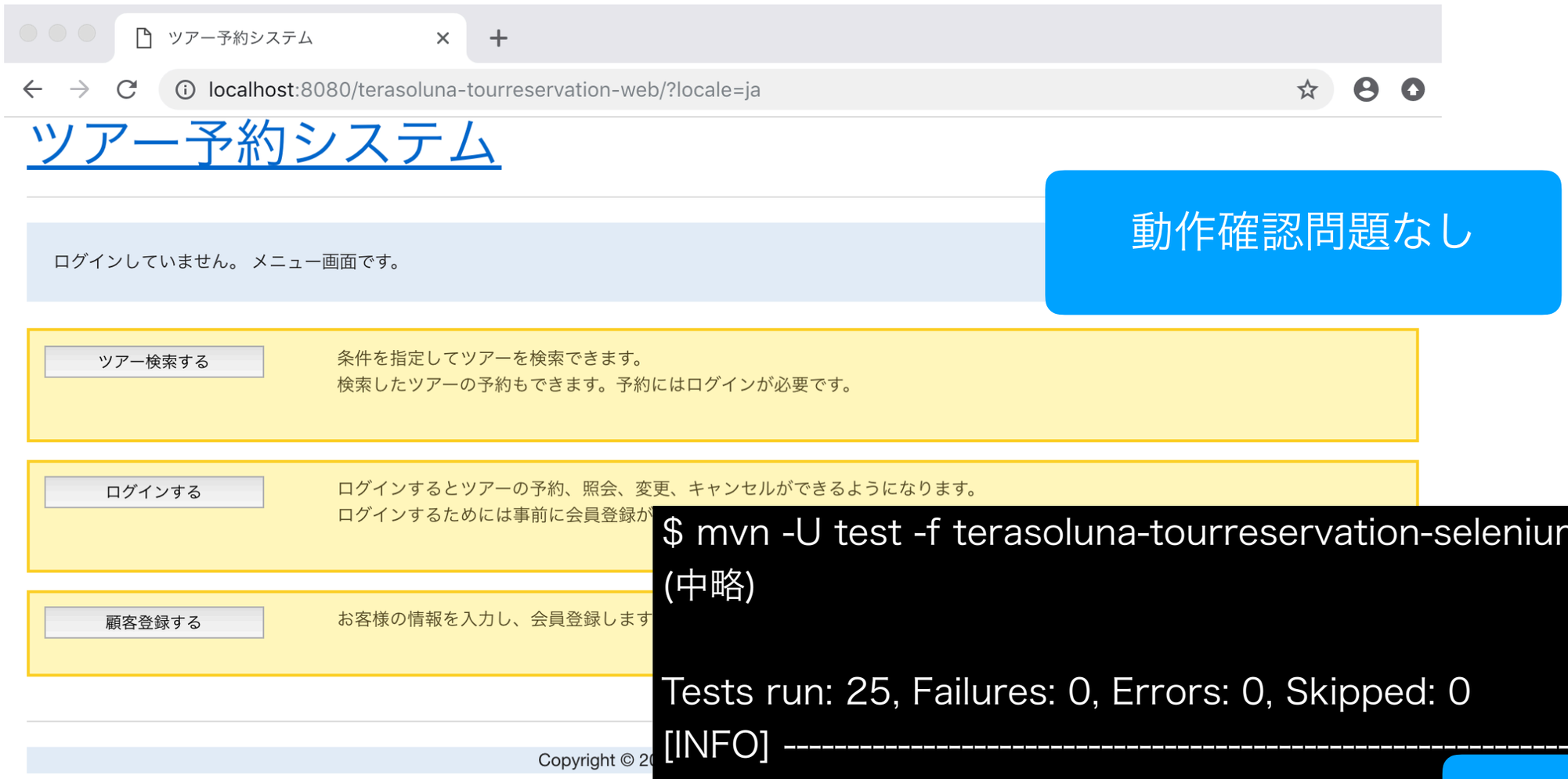
- JUnitで謎の実行時エラー

```
$ mvn install
(中略)

Caused by: java.lang.IllegalArgumentException: error the @within pointcut expression is only supported at Java
5 compliance level or above
        at org.aspectj.weaver.tools.PointcutParser.parsePointcutExpression(PointcutParser.java:301)
        at
org.springframework.aop.aspectj.AspectJExpressionPointcut.buildPointcutExpression(AspectJExpressionPoint
cut.java:217)
        at
org.springframework.aop.aspectj.AspectJExpressionPointcut.checkReadyToMatch(AspectJExpressionPointcut.
java:190)
(中略)
```

AspectJ周りが怪しいので、VerUP
1.18.10->1.8.13

エラー解消の末




```
$ mvn -U test -f terasoluna-tourreservation-selenium/pom.xml
(中略)

Tests run: 25, Failures: 0, Errors: 0, Skipped: 0
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

SeleniumもPass

苦労の思い出

ビルド履歴

https://travis-ci.org/ikeyat/terasoluna-tourreservation-mybatis3/builds

修正履歴

https://github.com/ikeyat/terasoluna-tourreservation-mybatis3/pull/3

その他気づき

- Javadoc生成の検査が厳しくなった

```
$ mvn javadoc:javadoc
(中略)
[ERROR] Exit code: 1 - /Users/ikeyat/Documents/dev/git/terasoluna-tourreservation-mybatis3/terasoluna-tourreservation-web/src/main/java/org/terasoluna/tourreservation/app/common/constants/MessageId.java:399: エラー: '>'の使用が不正です
[ERROR]       * <strong>label.tr.managecustomer.createFormMessage = <strong>Membership information input</strong> -> Confirm information
(中略)
[ERROR] /Users/ikeyat/Documents/dev/git/terasoluna-tourreservation-mybatis3/terasoluna-tourreservation-web/src/main/java/org/terasoluna/tourreservation/app/common/constants/MessageId.java:464: エラー: インライン要素<strong>内で使用できないブロック要素です: p
[ERROR]       * <p>
(中略)
[ERROR] /Users/ikeyat/Documents/dev/git/terasoluna-tourreservation-mybatis3/terasoluna-tourreservation-web/src/main/java/org/terasoluna/tourreservation/app/common/constants/MessageId.java:717: エラー: 自己終了要素は使用できません
[ERROR]       * <strong>label.tr.searchtour.reserveScreenTitleMessage = Please check the reservation details<br />
```

Spring4アプリを動かして学んだ

- Java非互換情報からの体系的な調査、処置を事前に実施しておかないと辛い
- 依存ライブラリやツールのVerUP必要性の判別が難しい。エラーが出たらVerUPを試すの繰り返し、行き当たりばったり(依存ライブラリVer組み合わせグジャグジャ)
- 回帰テストが信頼できないと、本当に対応しきれている確証が得られない（例：本当にAOPは全部動いてる？）
- 問題が見つかって解決策がなく取り残された時の絶望感

Agenda

- JavaのVer追従の必要性
- JavaのVer追従の課題
- Java8⇔Java11の非互換概要
- JavaライブラリやSpringのJava11サポート状況
- Spring4アプリケーションをJava8->11に対応させてみる（非推奨）
- まとめ

まとめ

- Java最新への追従は重要だが容易ではない（これまでも＋これからはより一層）
- Java上のライブラリ類のライフサイクル追従も同様（Spring然り）
- 悪魔の実を食べるような中途半端は極力避けた方が良い（依存ライブラリの崩れ）

繰り返さぬため心がけること

- 最新のJava等に追従することを想定したシステム構想をシステム所有者が考える（or 考えさせる）
- システム自体の長期ロードマップ計画(追従v.s.捨て)
- CI/CD、Dev/Ops(機動性を高める)
- 初期開発v.s.維持保守の期間・コスト配分のバランス

番外

Spring5.0アプリを動かしてみる

- TERASOLUNA Framework ver.5が公開するサンプルアプリ(tourreservation)のSpring5.0版（開発中）を使用
  - Spring 5.0 & MyBatis 3.4 @ Java8
  - JUnitテストや動作確認用Seleniumテスト付き

- OracleJDK11を無償範囲目的※で利用

※ https://www.oracle.com/technetwork/java/javase/terms/license/javase-license.html

Spring5.0

エラー遭遇その３

- JUnitでJAXBが見つからず実行時エラー

```
$ mvn install
(中略)

Tests run: 12, Failures: 0, Errors: 12, Skipped: 0, Time elapsed: 1.356 sec <<< FAILURE!
testReserve01(org.terasoluna.tourreservation.domain.service.reserve.ReserveServiceImplTest)  Time elapsed:
1.106 sec  <<< ERROR!
com.github.dozermapper.core.MappingException: org.xml.sax.SAXException: Implementation of JAXB-API has
not been found on module path or classpath.
javax.xml.bind.JAXBException: Implementation of JAXB-API has not been found on module path or classpath.
 - with linked exception:
[java.lang.ClassNotFoundException: com.sun.xml.internal.bind.v2.ContextFactory]
        at com.github.dozermapper.core.util.MappingUtils.throwMappingException(MappingUtils.java:78)
        at com.github.dozermapper.core.builder.xml.BeanMappingXMLBuilder.load(BeanMappingXMLBuilder.java:
119)
(中略)
```

Java11で完全削除されたJAXBをDozer6が利用していたため。JAXB置き換え先のライラリを追加する必要がある（次頁）。

Spring5.0

エラー遭遇その３

- JAXBの置き換えライブラリの追加例

pom.xml

```
<dependency>
    <groupId>com.sun.xml.bind</groupId>
    <artifactId>jaxb-impl</artifactId>
    <version>2.2.11</version>
</dependency>
<dependency>
    <groupId>com.sun.xml.bind</groupId>
    <artifactId>jaxb-core</artifactId>
    <version>2.2.11</version>
</dependency>
```

JDKに内蔵されなくなったため、
外部から調達するようにする
(旧JavaEE、現JakartaEE)

バージョンは一旦仮で選ぶ

Spring5.0

エラー遭遇その４

- JUnitでJAFが見つからず実行時エラー

```
$ mvn install
(中略)

Tests run: 12, Failures: 0, Errors: 12, Skipped: 0, Time elapsed: 1.502 sec <<< FAILURE!
testReserve01(org.terasoluna.tourreservation.domain.service.reserve.ReserveServiceImplTest)  Time elapsed:
1.171 sec  <<< ERROR!
java.lang.NoClassDefFoundError: javax/activation/DataSource
        at com.sun.xml.bind.v2.model.impl.RuntimeBuiltinLeafInfoImpl.<clinit>(RuntimeBuiltinLeafInfoImpl.java:470)
        at com.sun.xml.bind.v2.model.impl.RuntimeTypeInfoSetImpl.<init>(RuntimeTypeInfoSetImpl.java:63)
        at com.sun.xml.bind.v2.model.impl.RuntimeModelBuilder.createTypeInfoSet(RuntimeModelBuilder.java:
128)

(中略)
```

Java11で完全削除されたJAFをJAXBが利用していたため。
JAXBでのJAF利用はオプション扱いのため、
Mavenの依存関係により自動解決してくれない

Spring5.0

エラー遭遇その４

- JAFの置き換えライブラリの追加例

pom.xml

```
<dependency>
    <groupId>javax.activation</groupId>
    <artifactId>javax.activation-api</artifactId>
    <version>1.2.0</version>
</dependency>
```

JDKに内蔵されなくなったため、
外部から調達するようにする
(旧JavaEE、現JakartaEE)

バージョンは一旦仮で選ぶ

Spring5.0

エラー遭遇その５

- JUnitで謎のNullPointerException

```
$ mvn install
(中略)

Tests run: 7, Failures: 0, Errors: 1, Skipped: 0, Time elapsed: 3.37 sec <<< FAILURE!
testUpdateForOwnerReservation(org.terasoluna.tourreservation.domain.service.reserve.ReserveServiceImplSec
urityTest)  Time elapsed: 0.032 sec  <<< ERROR!
java.lang.NullPointerException
        at
org.terasoluna.tourreservation.domain.service.tourinfo.PriceCalculateSharedServiceImpl.calculatePrice(PriceCal
culateSharedServiceImpl.java:47)

(中略)
```

アプリケーションのビジネスロジック内で発生しているので、発生箇所のソースコードを見てみる
PriceCalculateSharedServiceImpl.java L47

Spring5.0

エラー遭遇その５

- JUnitで謎のNullPointerException

PriceCalculateSharedServiceImpl.java

```
(中略)
private Age adultAge;
private Age childAge;

public PriceCalculateOutput calculatePrice(Integer basePrice,
        Integer adultCount, Integer childCount) {

    PriceCalculateOutput result = new PriceCalculateOutput()
    int adultUnitPrice = basePrice * adultAge.getAgeRate() / 100;  // エラー発生行

(中略)
```

怪しい・・・

Spring5.0

エラー遭遇その５

- JUnitで謎のNullPointerException

PriceCalculateSharedServiceImpl.java

```
import javax.annotation.PostConstruct;

    (中略)
@PostConstruct
public void loadAges() {
        (中略)
        childAge = ageRepository.findOne("1");
        adultAge = ageRepository.findOne("0");
        (中略)
```

SpringDIコンテナ初期化時に呼び出す目印としてJavaEEのアノテーションが使われている。このJavaEEアノテーションはJAX-WSと共にava11で削除されたため、Springが検知できなかった模様。（なぜSpring4の時のようにコンパイルエラーにならないかは不明）

Spring5.0

Spring4.3

エラー遭遇その５

- JavaEEアノテーションの置き換えライブラリの追加例

pom.xml

```
<dependency>
    <groupId>javax.annotation</groupId>
    <artifactId>javax.annotation-api</artifactId>
    <version>1.3.1</version>
</dependency>
```

JDKに内蔵されなくなったため、
外部から調達するようにする
(旧JavaEE、現JakartaEE)

バージョンは一旦仮で選ぶ