フレームワーク基礎実習

02. Spring Boot 入門

株式会社ジードライブ

今回学ぶこと

- Springの概要
- ビルドツール(Maven)について
- プロジェクトの作成方法

Springとは

- Javaによるシステム開発を効率的に行うためのフレームワーク
  - 多様なシステムに対応できるよう Spring 〇〇〇という名称の様々なプロジェクトが存在する
  - https://spring.io/projects
- Spring Frameworkは、Springプロジェクトの中核を担う
  - アプリケーション開発に必要な機能を包括的に含み、他のSpringプロジェクトや他のフレームワークと連携することができる
- Spring Bootは、Spring Frameworkを拡張したもの
  - Spring Frameworkで必要だった煩雑な設定することなく、簡単に利用することができる

Springの提供するおもな機能

- DIコンテナ
  - DI(Dependency Injection)：オブジェクトの生成／オブジェクト同士の連携を制御する仕組み
- AOPコンテナ
  - AOP(Aspect Oriented Programming)：オブジェクト本来の責務とそれ以外のロジックを切り離し、後から追加できるようにする仕組み
- Spring Web
  - MVCパターンのWebアプリケーションを効率的に開発するための仕組み
- 他のフレームワークとの連携機能
  - 他のフレームワーク(MyBatis, Hibernate等)との連携が容易

Springの特徴：アノテーション

- Springを使用して、システムを開発する際は、
  アノテーションを多用する
  - アノテーションによって、クラスやフィールド等に目印を付け、
    その目印に応じて、Springがクラスやフィールドを利用する

アノテーションの例

```
@Bean, @Component, @Autowired
@Configuration, @Controller, @Service
@Repository, @Valid, @RequestMapping …etc
```

  - 例えば、@Configurationというアノテーションが付いているク
    ラスは、設定ファイルとしてSpringに利用される

アノテーションの利用例

- @Controllerというアノテーションによって、HelloControllerはブラウザからのリクエストに応えることができるようになる
  ⇒ @GetMappingで指定されたURLに対応して、sendHello()メソッドが実行される
- @Componentによって、MailServiceオブジェクトが生成され、@Autowiredの付与されたserviceフィールドに注入される

HelloController.java

```
@Controller
public class HelloController {
  @Autowired
  private MailService service;

  @GetMapping("/hello")
  public String sendHello() {
    service.send("こんにちは");
    …
  }
}
```

MailService.java

```
@Component
public class MailService {

  public void send(String msg) {
    // メール送信処理
    var mail = new SimpleMail();
    …
  }

}
```

Springの導入方法

- マニュアルで必要なJARファイルをダウンロードして利用することも可能だが、一般的にはMavenやGradleといった管理ツールを利用する
- Spring Bootを利用する場合は、以下のWebページ(Spring Initializr)からひな形となるプロジェクトファイルをダウンロードして、Eclipseにインポートする
  - https://start.spring.io/

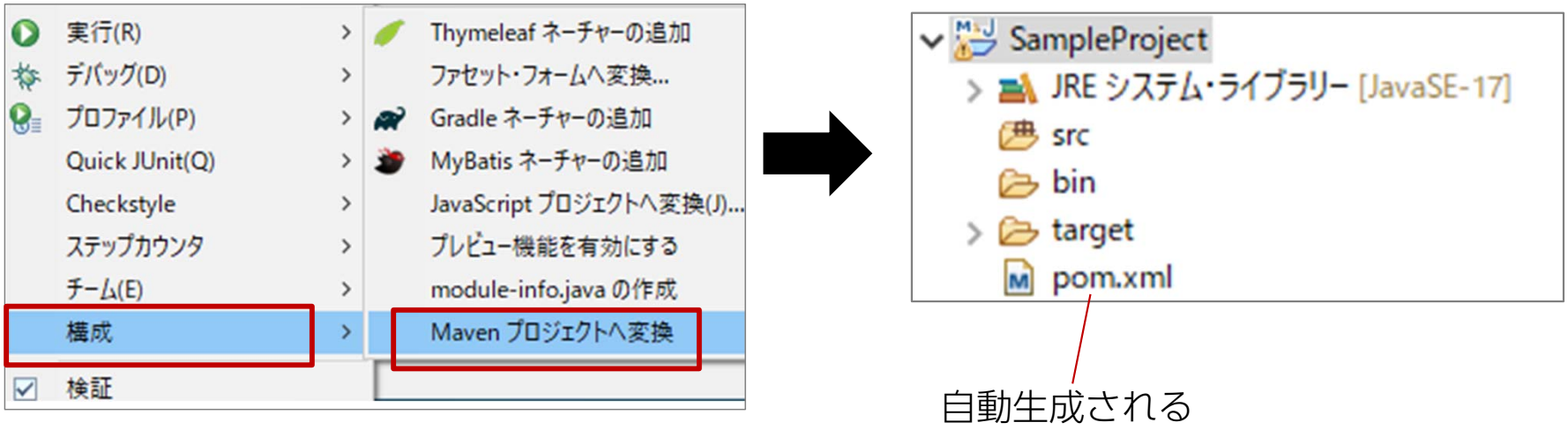
ビルドツール (Maven)

ビルドツールの利用

- ライブラリやフレームワークを利用する際は、必要なJARファイルをダウンロードし、自身のプロジェクト内で使えるように準備する
  - 利用したいライブラリが、他のライブラリに依存している場合、別途JARファイルを準備する必要がある
  - 同プロジェクトの作業を、別の環境で行う場合、改めてJARファイルを準備する必要が生じる
- この作業は手動で行うのではなく、MavenやGradleといったビルドツールを利用することが一般的
  - ライブラリの管理だけでなく、デプロイ用のJARファイルをビルドすることができる

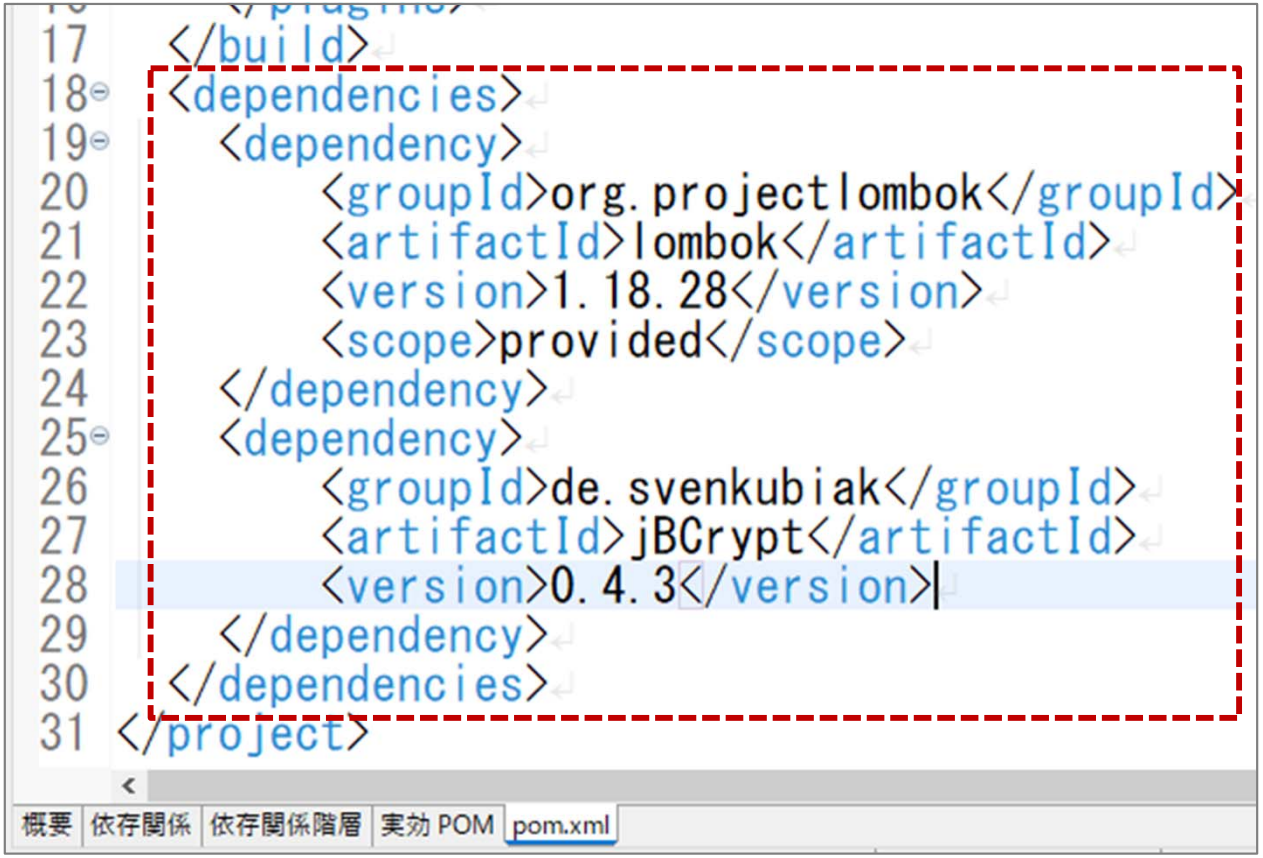
Mavenの利用方法

- Eclipseでプロジェクトフォルダを右クリックし、「構成 ⇒ Mavenプロジェクトへ変換」する
  - pom.xmlというファイルが生成され、Maven管理のプロジェクトになる



pom.xmlの記述

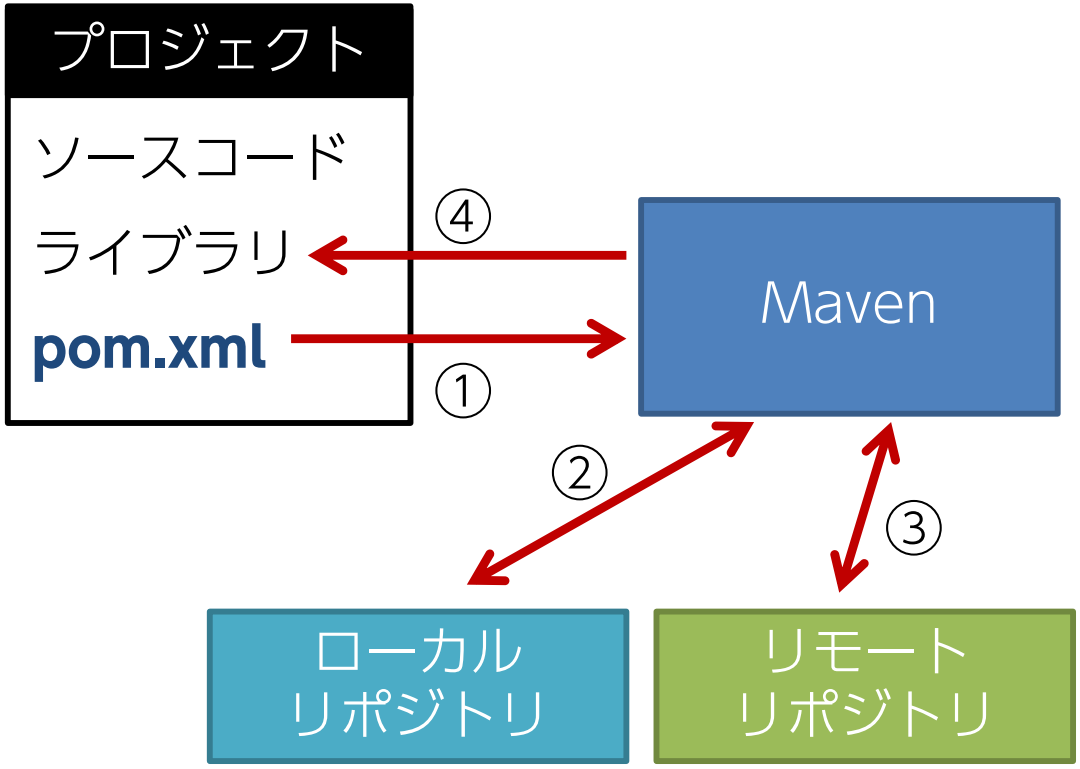
- pom.xmlには、プロジェクトについての基本情報や使用するライブラリについての記述を行う
  - 使用するライブラリの情報は、dependencies要素内に記述する



Mavenによるライブラリの管理

- Mavenはpom.xmlに従い、必要なライブラリをダウンロードして、利用可能な状態にする

① pom.xmlに必要なライブラリを記載
② ローカルリポジトリをチェック
③ ローカルにない場合、リモートリポジトリから取得し、 ローカルに保存
④ ライブラリとして使用可能になる



C:¥Users¥zdXXXX¥.m2 内にダウンロードされる

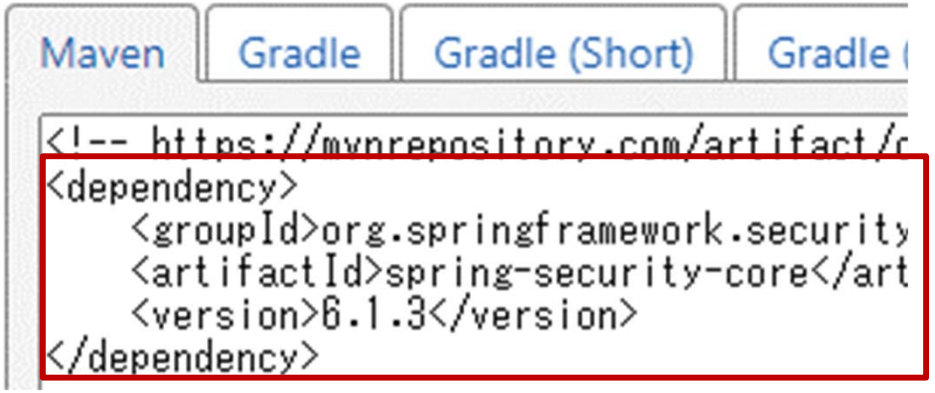
Maven Repositoryの利用

- 使用するライブラリについては、Maven Repository で検索する（https://mvnrepository.com/）

② 目的のものをクリック

Maven Repositoryの利用

- 利用するバージョンを選び、dependency要素をコピー
  ⇒ pom.xmlに貼り付ける

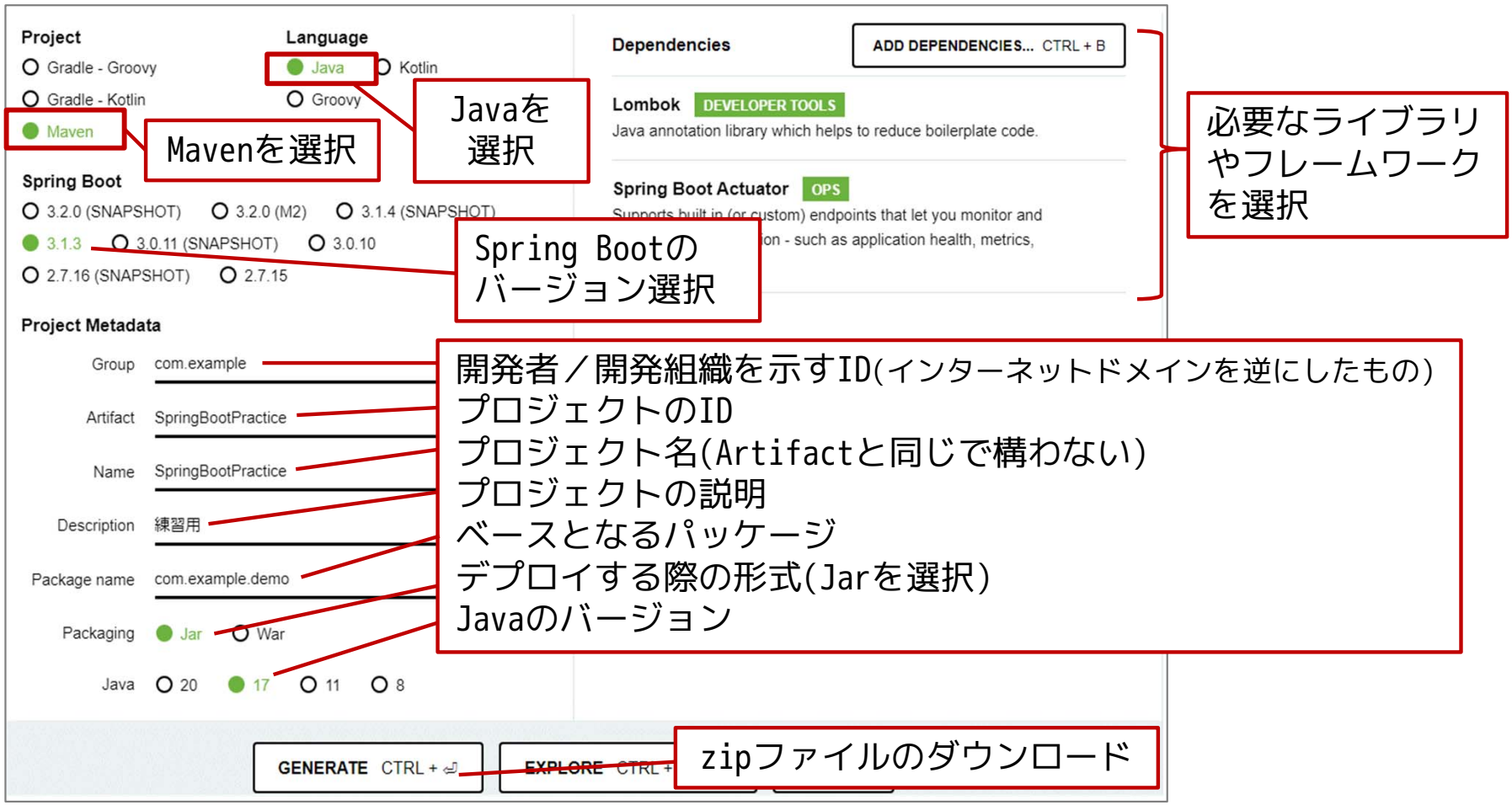


groupId：開発者／開発組織を示すID
artifactId：プロダクトのID
version：プロダクトのバージョン

Spring Bootプロジェクトの作成

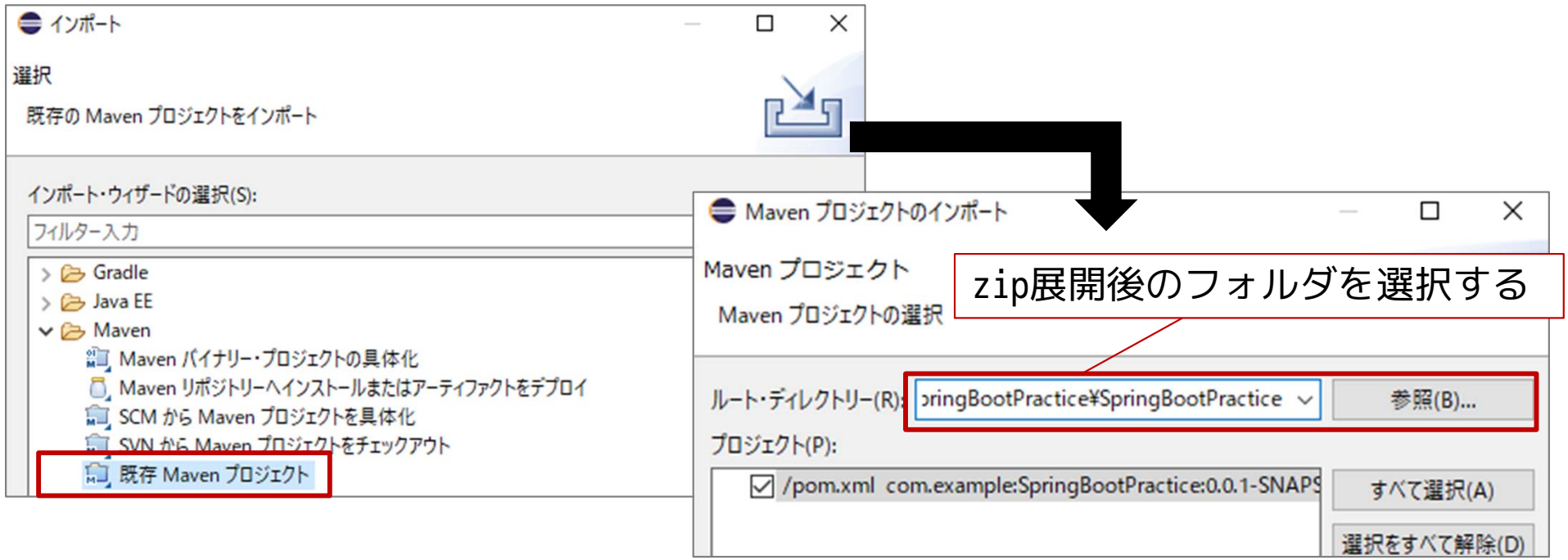
Spring Bootプロジェクトの作成

1. https://start.spring.io/ で必要な項目を設定して、zipファイルをダウンロード



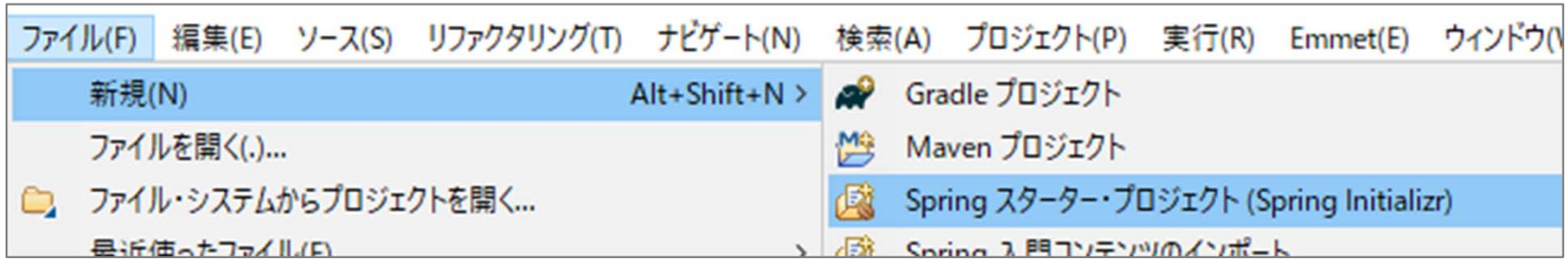
Spring Bootプロジェクトの作成

2. zip展開後のフォルダは、Eclipseで「ファイル ⇒ インポート」メニューから「既存Mavenプロジェクト」としてインポートする
   - 教材として配布しているサンプルのプロジェクトも、同様の方法でEclipseに読み込むことができる



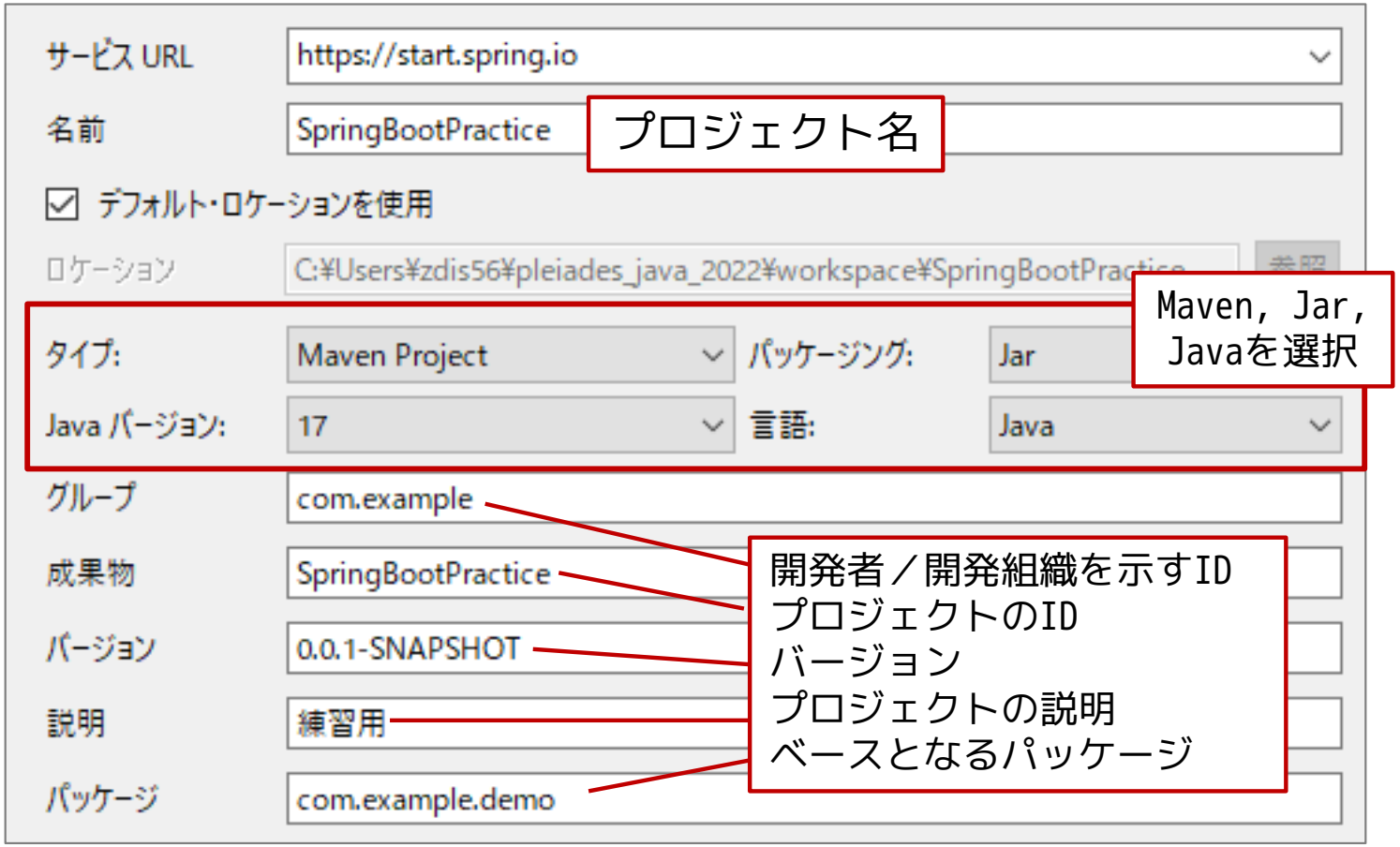
Eclipseでのプロジェクト作成

- Eclipseでは、「Springスターター・プロジェクト」を作成することで、start.spring.ioからのひな形のダウンロードとインポートが行われる



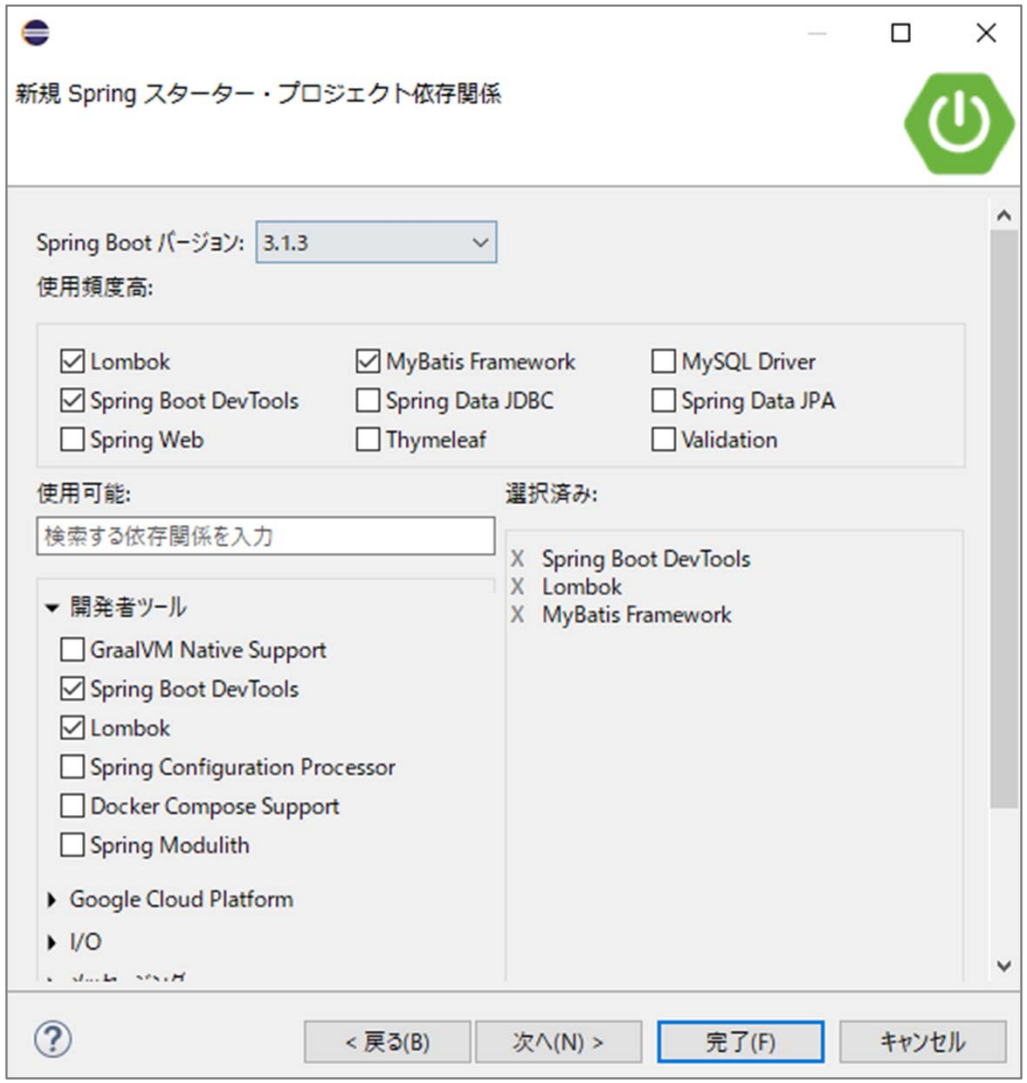
Eclipseでのプロジェクト作成

- ファイル ⇒ 新規 ⇒ Spring スターター・プロジェクト



Eclipseでのプロジェクト作成

- プロジェクトに必要なツール(依存関係)を選択する
  - 選択したものが`pom.xml`に記載される



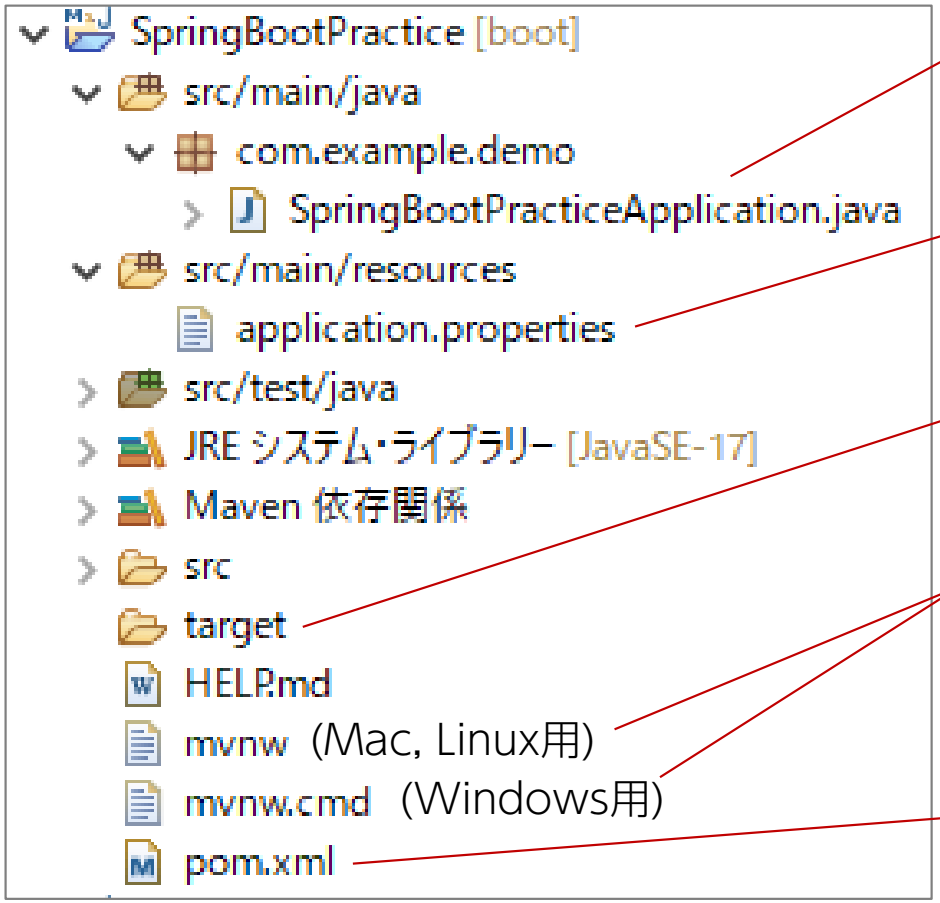
おもな依存関係

- 訓練では以下のツール(依存関係)を使用する

カテゴリ	依存関係	説明
Web	Spring Web	SpringのMVC機能を利用できる
テンプレート・エンジン	Thymeleaf	JSPの代替となるテンプレート ファイル拡張子は、.html
開発ツール	Spring Boot DevTools	ファイル更新時のTomcat再起動などを自動化する
開発ツール	Lombok	コンストラクタやアクセッサの記述を簡略化することができる
I/O	Validation	入力値のバリデーションをアノテーションによって行うことができる
SQL	MySQL Driver	データベース接続に必要
SQL	MyBatis Framework	データベースとの連携を行う際に利用するのSQLマッパー

プロジェクトの構成

- プロジェクト作成後は以下のような構成になる



プロジェクト作成時に自動生成される
クラスでmainメソッドをもつ

データベース接続情報などを記載する
アプリケーションの設定ファイル

ビルド後のデータ(Jarファイル)が格納
されるフォルダ

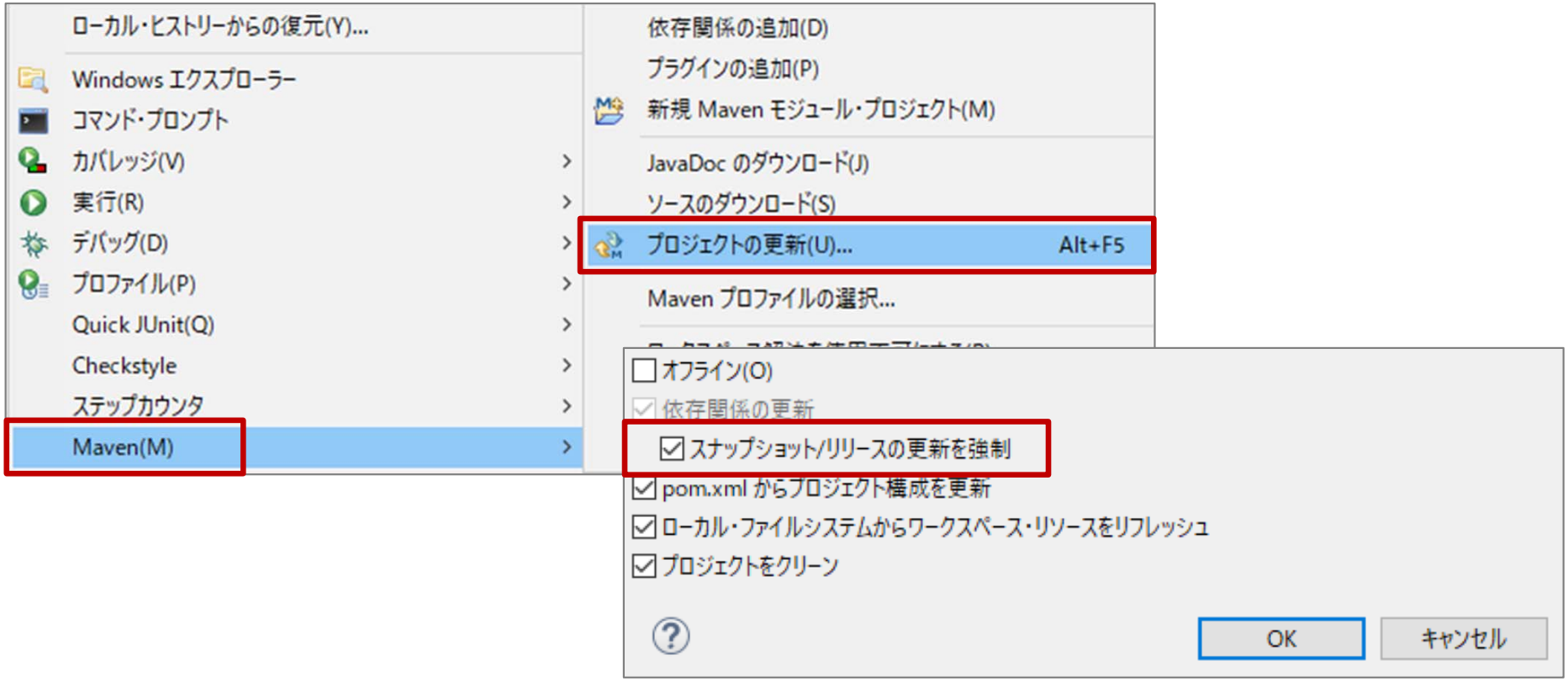
Maven Wrapper：
Mavenがインストールされていない環境
であってもMavenコマンドが利用できる

プロジェクトの情報や依存関係などが
記されたMaven用の設定ファイル

※ パッケージ・エクスプローラーでは表示されていないが、.gitignoreも生成される

pom.xmlのエラー

- プロジェクト作成時に、pom.xmlにエラーが生じる場合は、プロジェクト上で右クリックし、「Maven⇒プロジェクトの更新」「スナップショット/リリースの更新を強制」にチェックを入れて、更新する



依存関係の追加

- プロジェクト作成後に依存関係を追加する場合、プロジェクト上で右クリックし、「Spring ⇒ スターターの追加」から追加する依存関係を選択する
  - 「次へ」進んで、構造体の比較のpom.xmlにチェックを入れて、完了する