

# spring-sample

---

Spring bootのサンプルプロジェクトです。

RESTFulなAPIでデータベースにアクセスしてデータの更新、取得が行えます。

## 前提条件

- eclipse にLombokが導入済み
- java8以上がインストール済み
- postgresqlがインストール済み

LombokについてはpleiadesのLombok付きをインストールすると楽[pleiades](#)

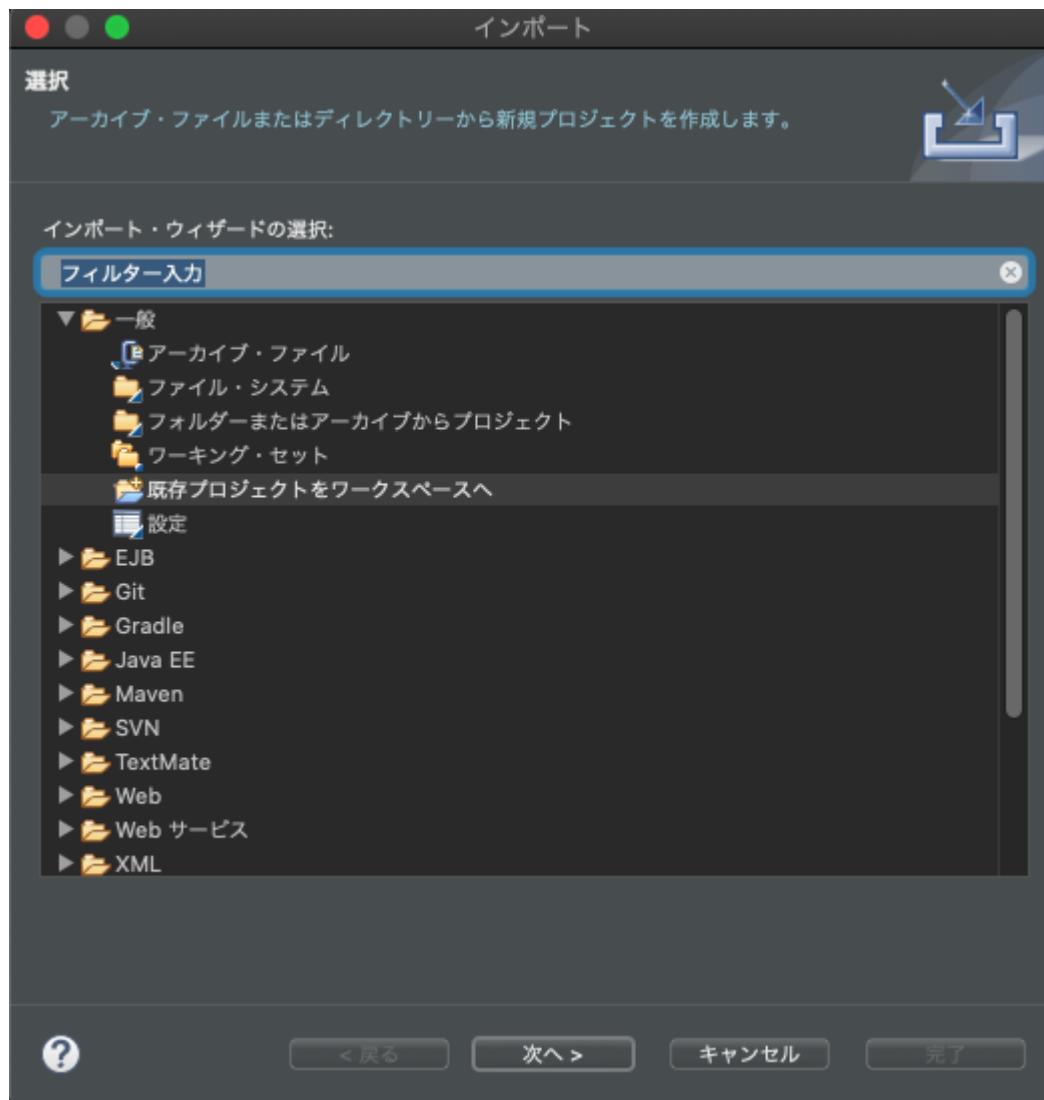
## 動作確認環境

環境	サービス/バージョン
実行環境	Mac OS Catalina、Windows10
開発環境	eclipse pleiades 2019
開発言語	Java 8
DB	PostgreSQL 10
Framework	SpringBoot 2.2.6
依存関係	Jersey、MyBatis、Jackson

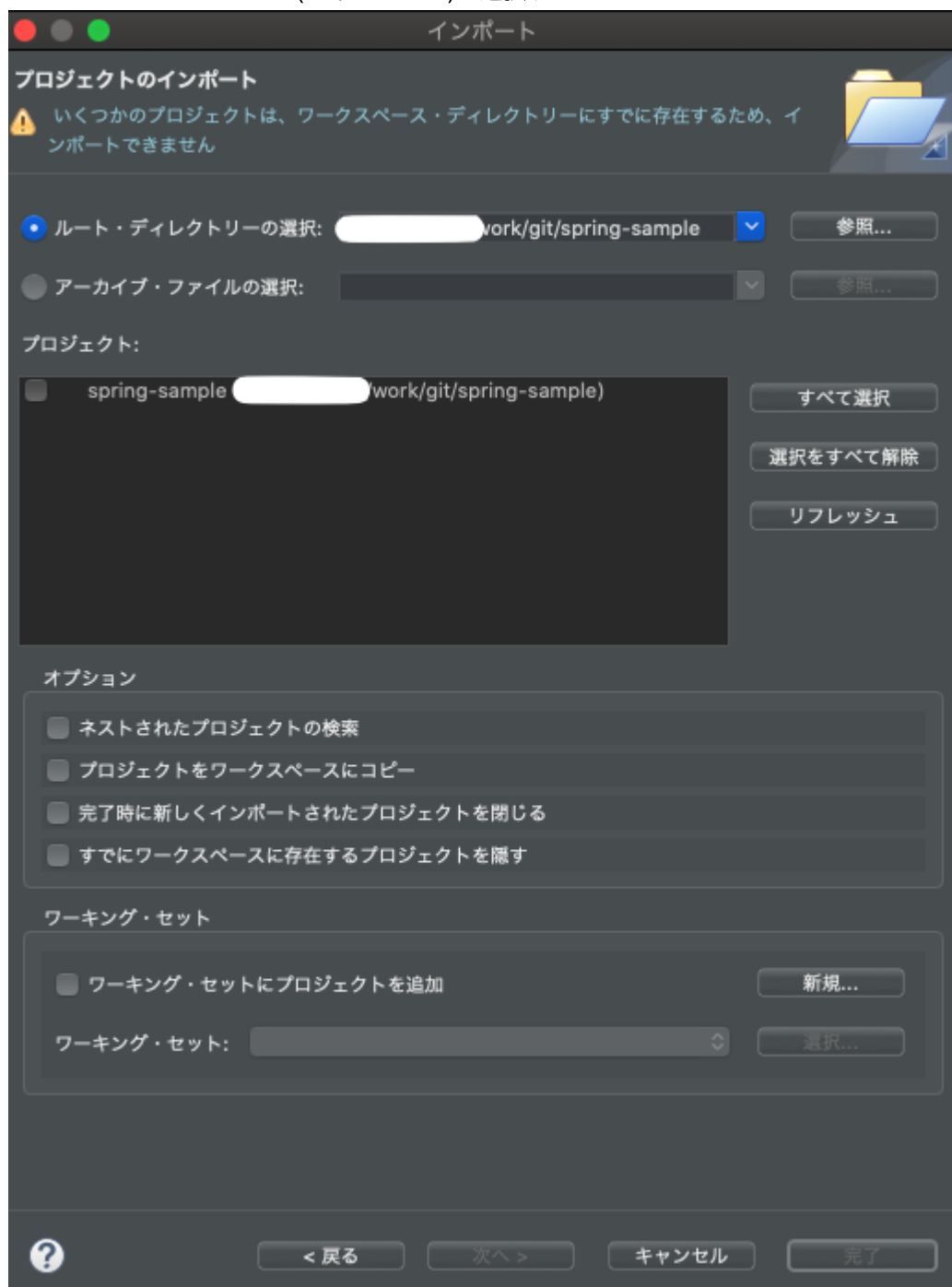
## 手順

1. リポジトリのクローン  
Gitのクライアントツール等でローカルにこのリポジトリをクローンする  
eclipseでインポートするためクローンする場所はメモしておく
2. eclipseにプロジェクトのインポート  
クローンしたリポジトリをeclipseでインポートする

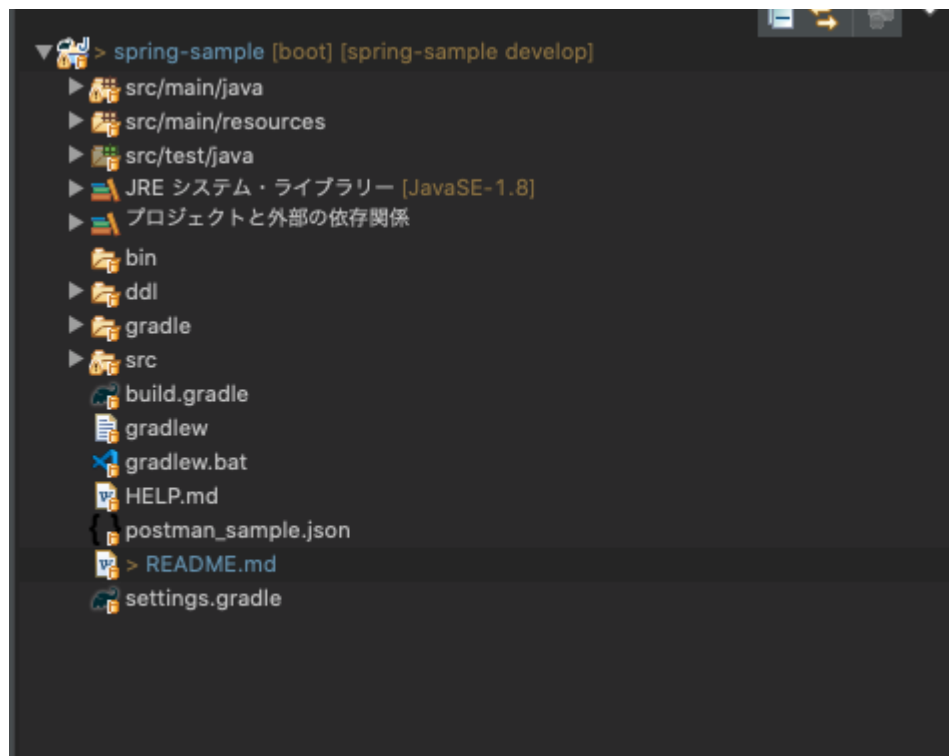
一般->既存プロジェクトをワークスペースへ



クローンしたプロジェクト(ディレクトリ)を選択する

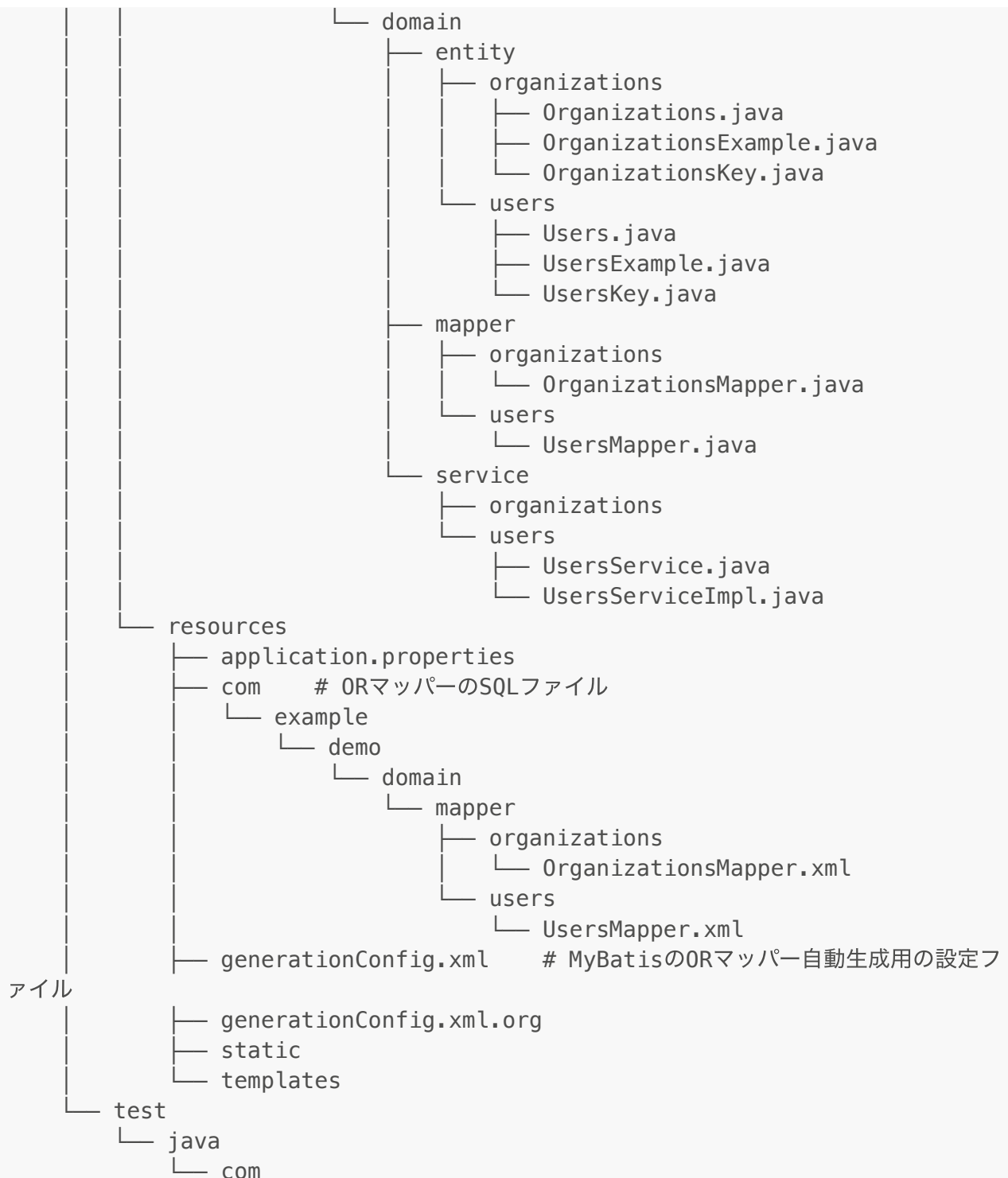


インポート完了



主なリソースの構成

```
├── build.gradle # gradleの依存関係をまとめるファイル
├── ddl # テスト用のDB生成DDL
│   ├── 01.role.sql
│   ├── 02.database.sql
│   └── 03.tables.sql
├── gradle # gradle関連ファイル(特に使用しない)
│   └── wrapper
│       ├── gradle-wrapper.jar
│       └── gradle-wrapper.properties
├── gradlew # gradleバッチファイル、コマンドラインでビルド等する際に使用
├── gradlew.bat # gradleバッチファイル、コマンドラインでビルド等する際に使用
├── postman_sample.json # 試験用ツールのテンプレートPostman
├── settings.gradle # gradle設定ファイル
└── src # プログラムファイル
    ├── main
    │   ├── java
    │   │   ├── com
    │   │   │   ├── example
    │   │   │   │   ├── demo
    │   │   │   │   │   ├── SpringSampleApplication.java
    │   │   │   │   │   ├── api
    │   │   │   │   │   │   ├── organizations
    │   │   │   │   │   │   └── users
    │   │   │   │   │   │       ├── UserRestController.java
    │   │   │   │   │   │       └── UsersBean.java
    │   │   │   │   └── config
    │   │   │   │       ├── JerseyConfig.java
    │   │   │   │       └── WebConfig.java
```



ORマッパーの自動生成のやり方は[こちら](#)を参照

### 3. テスト用データベース作成

クローンしたリポジトリ内にあるDDLを以下の順番で実行してデータベースを作成する

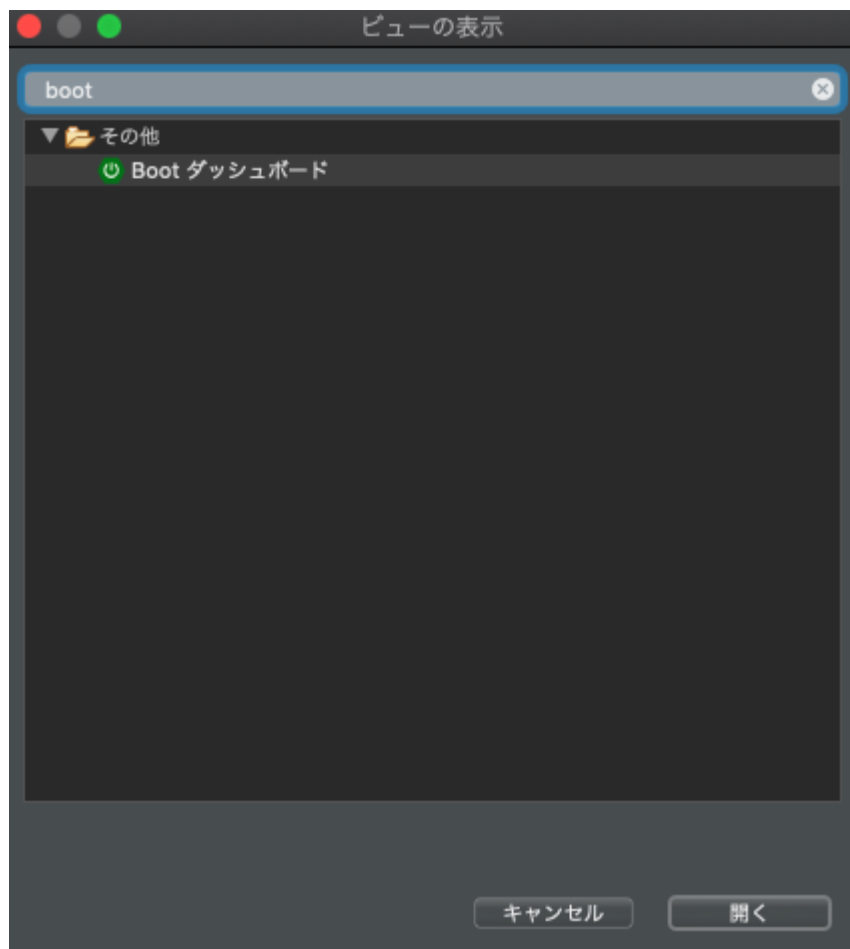
- 01.role.sql
- 02.database.sql
- 03.tables.sql

ユーザパスワードはspring/spring

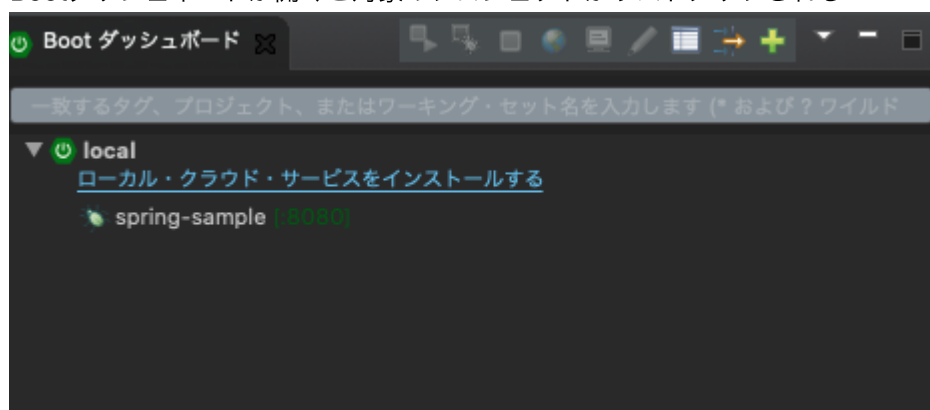
#### 4. アプリケーションの起動

eclipseを用いたアプリケーションの起動

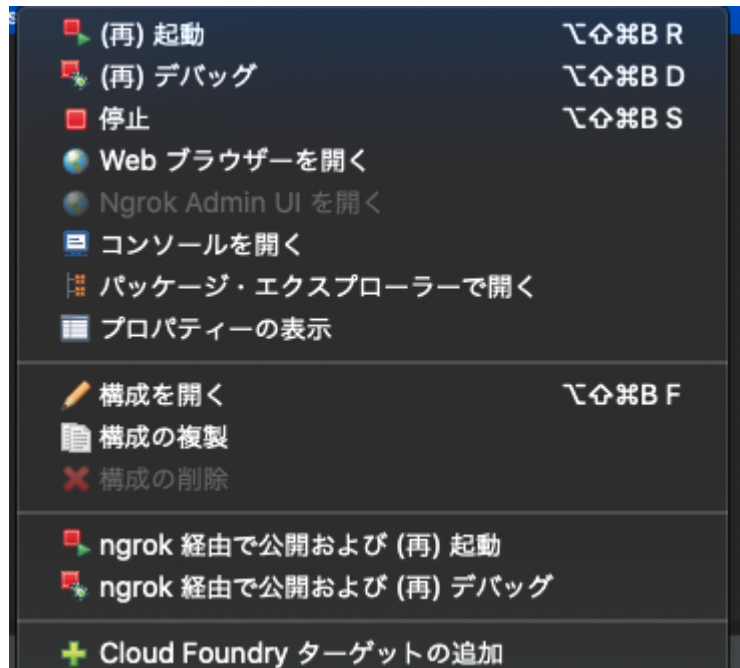
ビューの表示でBootダッシュボードを表示する



Bootダッシュボードが開くと対象のプロジェクトがリストアップされる



右クリックでデバッグ、もしくは通常起動をクリックするとアプリが起動する



以下のコンソールが出力されると起動完了

Started SpringSampleApplication in 1.821 seconds

```

Spring Boot (v2.2.6.RELEASE)

2020-04-21 10:43:06.895 INFO 39935 --- [main] c.example.demo.SpringSampleApplication : Starting SpringSampleApplication on naotonoMacBook-Pro.local with PID 39935
2020-04-21 10:43:06.898 INFO 39935 --- [main] c.example.demo.SpringSampleApplication : No active profile set, falling back to default profiles: default
2020-04-21 10:43:07.728 INFO 39935 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2020-04-21 10:43:07.735 INFO 39935 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-04-21 10:43:07.735 INFO 39935 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.33]
2020-04-21 10:43:07.791 INFO 39935 --- [main] o.a.c.c.f.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-04-21 10:43:07.792 INFO 39935 --- [main] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in 850 ms
2020-04-21 10:43:08.193 INFO 39935 --- [main] o.s.b.w.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-04-21 10:43:08.436 INFO 39935 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2020-04-21 10:43:08.439 INFO 39935 --- [main] c.example.demo.SpringSampleApplication : Started SpringSampleApplication in 1.821 seconds (JVM running for 2.533)

```

##### 5. 動作確認 アプリが起動したらRESTクライアントツールを用いて動作確認を実施する

テストはChromeAppのPostmanを使用する(PostmanでなくてもCurl等のツールがあればそちらで問題無し)  
インストールは[こちら](#)を参照

Postmanを使用する場合はリポジトリにある以下の定義がインポートして使用可能です。

postman\_sample.json

インポートすると以下の定義が読み込まれます。

▼	tmp ☆	▶
	4 requests	⋮
GET	http://localhost:8080/spring-demo/v1/users/	
GET	http://localhost:8080/spring-demo/v1/users/{id}	
POST	http://localhost:8080/spring-demo/v1/users/	
PUT	http://localhost:8080/spring-demo/v1/users/{id}	

それぞれユーザの一覧取得、ID指定の取得、登録、更新です。

## 最後に

今回テーブルUsersとOrganizations 2つ用意し、サンプルのAPIとしてUsersがあります。

基本的にはUsersを参考にすればOrganizationsのAPIも作成できると思います。

手順ではeclipse上で動作確認していますが、gradleでjarを生成すればTomcatが同梱されたモジュールが生成できるので、スタンドアロンで動作させる事も可能です。モジュールの生成方法等はeclipseでやる方法やバッチファイルから実施する方法があるので、やりやすい方を調べて試してください。