

A faint, light blue world map is visible in the background of the slide, centered behind the text.

INTERNET ACADEMY

Institute of Web Design & Software Services

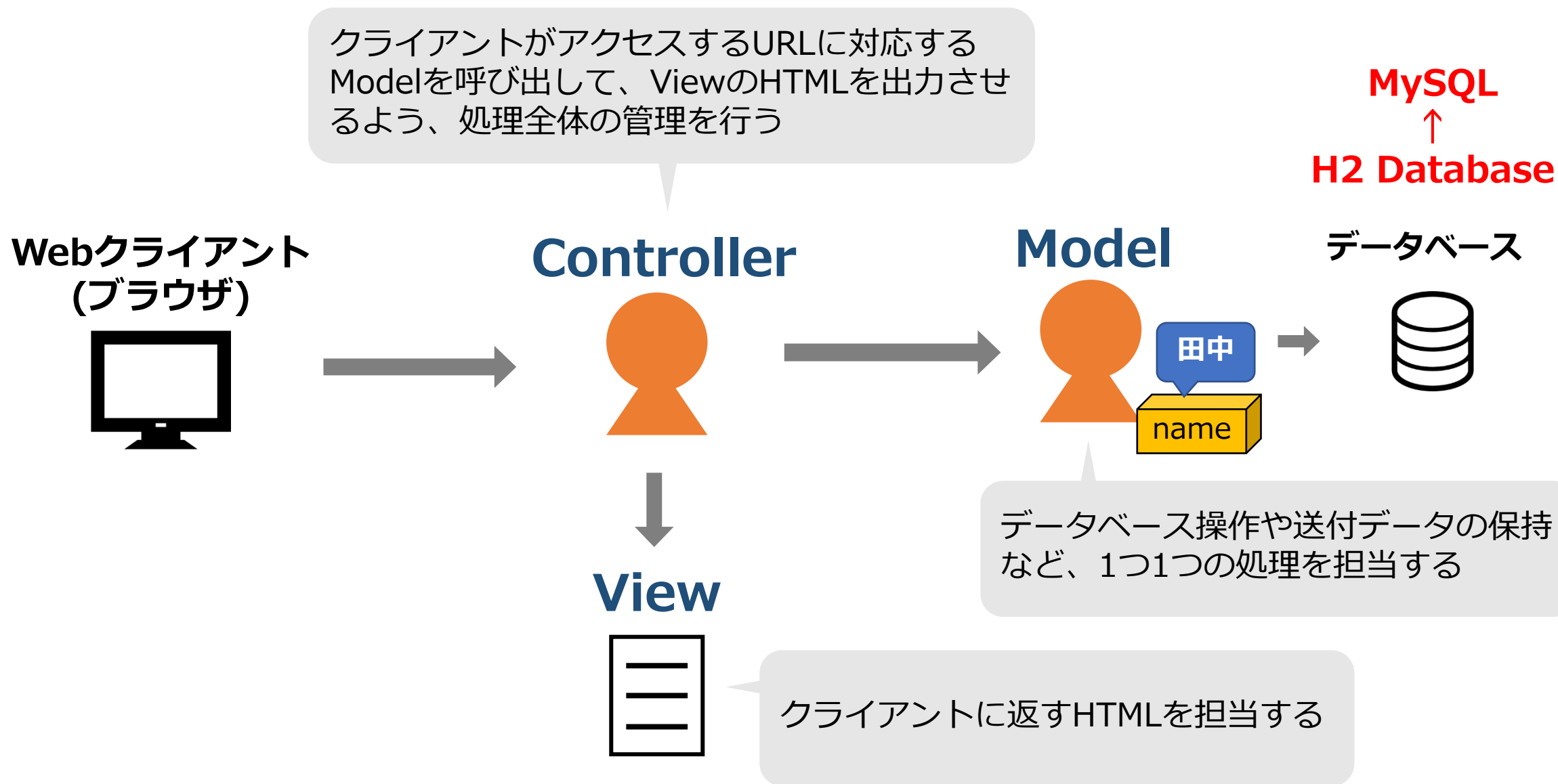
Spring Boot5

インターネット・アカデミー

Spring Boot 5 目次

- MySQLの利用
- JPAの利用(H2 database)
- JPAの利用(MySQL)
- JPAによるDB操作 1 (H2 database)

Spring MVCアーキテクチャとは



データベース接続時に必要なもの

項目	目的
Spring Boot DevTools	開発を便利にするツール (コード変更時の自動再起動など)
Thymeleaf	テンプレートエンジン
Spring Web	Spring MVCを使う
JDBC API	JDBCライブラリを使う
MySQL Driver	データベースにMySQLを使う



The screenshot shows the 'New Spring Starter Project Dependencies' window in an IDE. The 'Spring Boot Version' is set to 2.4.5. Under 'Usage Frequency', 'MySQL Driver' and 'Spring Web' are checked. In the 'Available' list on the left, 'Alibaba', 'Amazon Web Services', 'Development Tools', 'Google Cloud Platform', 'I/O', 'Messaging', 'Microsoft Azure', 'NoSQL', 'Observability', 'Ops', 'SQL', 'Security', 'Spring Cloud', 'Spring Cloud Circuit Breaker', and 'Spring Cloud Config' are listed. The 'Selected' list on the right includes 'Spring Boot DevTools', 'JDBC API', 'MySQL Driver', 'Thymeleaf', and 'Spring Web'. At the bottom, there are buttons for 'Default', 'Clear Selection', 'Back', 'Next', 'Finish', and 'Cancel'.

新規 Spring スターター・プロジェクト依存関係

Spring Boot バージョン: 2.4.5

使用頻度高:

☐ H2 Database ☒ JDBC API ☐ Lombok
☒ MySQL Driver ☒ Spring Boot DevTools ☐ Spring Data JPA
☒ Spring Web ☒ Thymeleaf ☐ 検証

使用可能: 選択済み:

検索する依存関係を入力

▶ Alibaba
▶ Amazon Web サービス
▶ 開発ツール
▶ Google Cloud Platform
▶ I/O
▶ メッセージング
▶ Microsoft Azure
▶ NoSQL
▶ Observability
▶ Ops
▶ SQL
▶ セキュリティー
▶ Spring Cloud
▶ Spring Cloud Circuit Breaker
▶ Spring Cloud Config

X Spring Boot DevTools
X JDBC API
X MySQL Driver
X Thymeleaf
X Spring Web

デフォルトにする 選択をクリア

? < 戻る(B) 次へ(N) > 完了(F) キャンセル

データベースの設定

application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/spb_db1?characterEncoding=UTF-8
```

```
spring.datasource.username=testuser
```

```
spring.datasource.password=testpass
```

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

#schema.sqlとdata.sqlを有効にするかどうか(初期値はembedded : 埋込DB(H2 Database)の時のみ実行)

#MySQLの場合は、実行時にリセットされないため、2回目以降でschema.sqlに工夫が必要。

```
spring.datasource.initialization-mode=always
```

データベースの設定

application.properties

DBの場所とポート番号

データベース名

`spring.datasource.url=jdbc:mysql://localhost:3306/spb_db1?characterEncoding=UTF-8`

`spring.datasource.username=testuser`

ユーザー名と
パスワード

`spring.datasource.password=testpass`

`spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver`

MySQL用のドライバ

#schema.sqlとdata.sqlを有効にするかどうか(初期値はembedded : 埋込DB(H2 Database)の時のみ実行)

#MySQLの場合は、実行時にリセットされないため、2回目以降でschema.sqlに工夫が必要。

`spring.datasource.initialization-mode=always`

データベースの初期化設定

データベースの設定

spring.datasource.initialization-modeの項目

値	説明
none	何もしない
validate	検証をしますが、データベースには変更を加えません。
update	アプリケーション起動時に、Entityに対応するテーブルがなければ作成します。
create	アプリケーション作成時に、Entityに対応するテーブルがなければ作成します。もしあれば、データを削除します。
create-drop	アプリケーション作成時に、Entityに対応するテーブルがなければ作成します。セッション終了時にスキーマを削除します。
embedded	埋込DB(H2 Database)を使っている時だけ、テーブルを作成します
always	常にテーブルを作成します

データベースの設定

schema.sql

```
--2回目以降で、DROP…を入れないとエラーになる
--DROP TABLE sample;
CREATE TABLE sample(
  id INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(100) NOT NULL,
  PRIMARY KEY(id)
);
```

data.sql

```
INSERT INTO sample (name) VALUES ('taro');
```

実行後の確認

phpMyAdminでの確認

← サーバ: 127.0.0.1 » データベース: spb_db1

構造 SQL 検索 クエリ エクスポート インポート 操作 権限 ルーチン イベント

フィルタ

含まれている文字:

テーブル	操作	行	タイプ	照合順序	サイズ	オーバーヘッド
<input type="checkbox"/> sample	★ 表示 構造 検索 挿入 空にする 削除	1	InnoDB	utf8_general_ci	16.0 KiB	-
1 テーブル	合計	1	InnoDB	utf8_general_ci	16.0 KiB	0 バイト

↑ ☐ すべてチェックする チェックしたものを:

印刷 データ辞書

テーブルを作成

名前: カラム数:

実行

実行後の確認

phpMyAdminでの確認

← サーバ: 127.0.0.1 » データベース: spb_db1 » テーブル: sample

☰

✓ 行 0 - 0 の表示 (合計 1, クエリの実行時間: 0.0023 秒。)

```
SELECT * FROM `sample`
```




☐ プロファイリング [[インライン編集](#)] [[編集](#)] [[EXPLAIN で確認](#)] [[PHP コードの作成](#)] [[更新](#)]


☐ すべて表示 | 行数: 25 | 行フィルタ: このテーブルを検索

+ オプション

← T →

	id	name
<input type="checkbox"/>  編集	1	taro

↑ ☐ すべてチェックする | チェックしたものを:  編集  コピー  削除

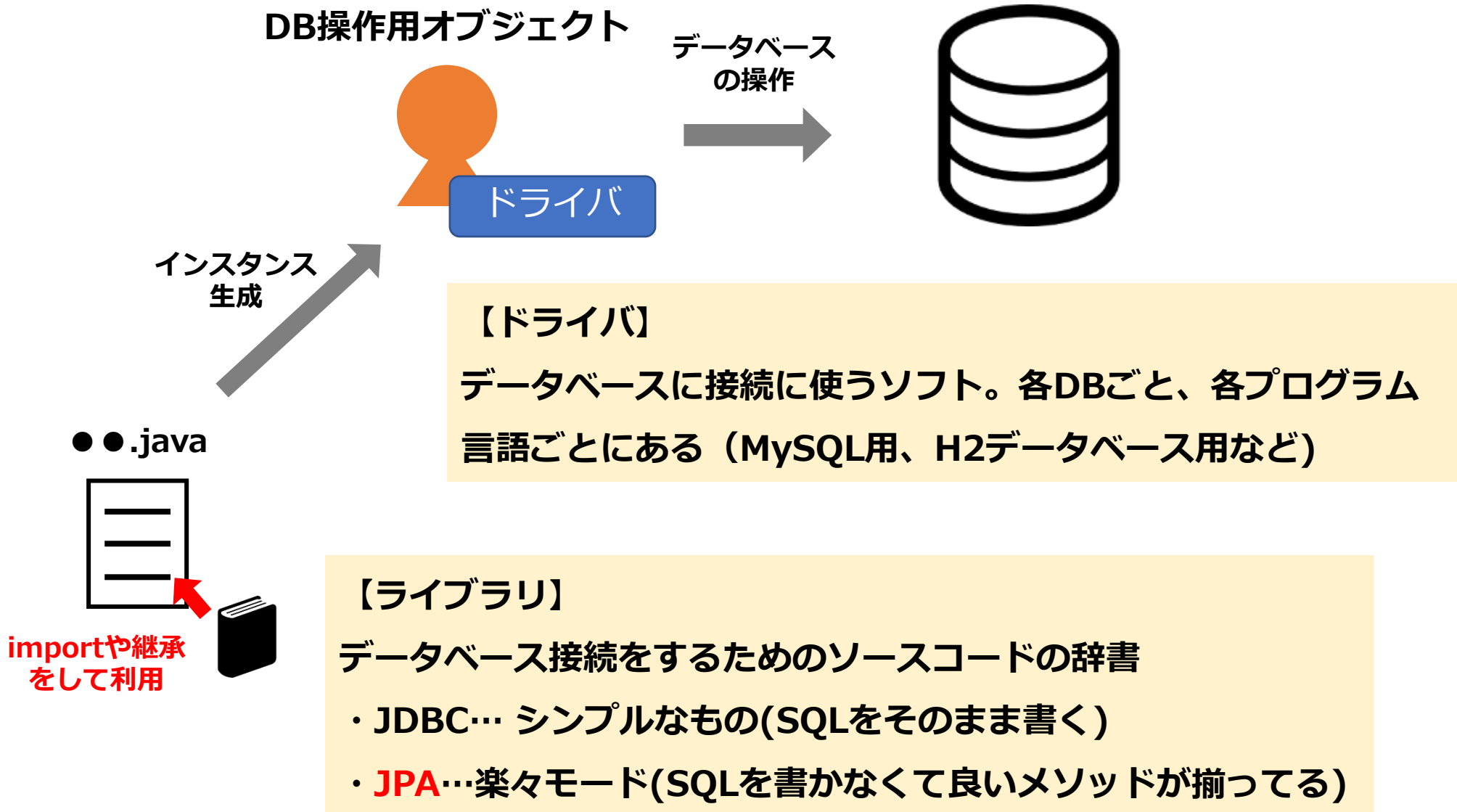
 エクスポート

☐ すべて表示 | 行数: 25 | 行フィルタ: このテーブルを検索

A faint, light blue world map is visible in the background of the slide, centered behind the title text.

JPAの利用(H2 database)

データベース接続時に必要なもの



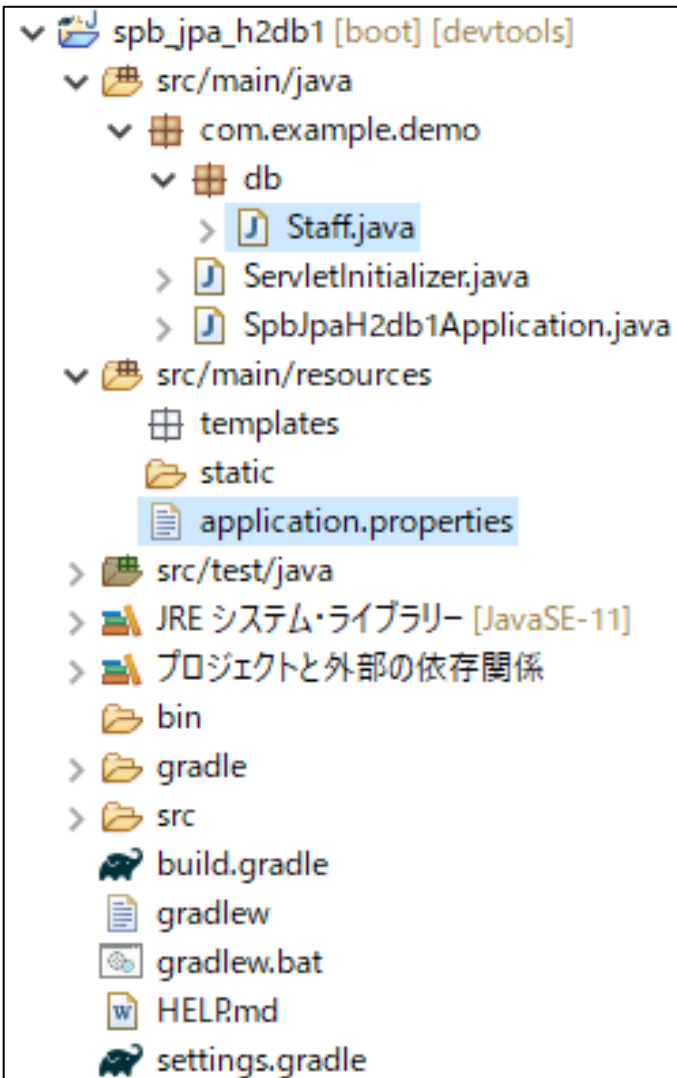
データベース接続時に必要なもの

項目	目的
Spring Boot DevTools	開発を便利にするツール (コード変更時の自動再起動など)
Thymeleaf	テンプレートエンジン
Spring Web	Spring MVCを使う
Spring Data JPA	JPAライブラリを使う
H2 Database	データベースにH2を使う



The screenshot shows the 'New Spring Starter - Project Dependencies' window. At the top, it says '新規 Spring スターター・プロジェクト依存関係'. Below this, 'Spring Boot バージョン:' is set to '2.4.5'. The '使用頻度高:' (High Frequency Use) section contains several checkboxes: ☒ H2 Database, ☐ MySQL Driver, ☒ Spring Web, ☐ JDBC API, ☒ Spring Boot DevTools, ☒ Thymeleaf, ☐ Lombok, ☒ Spring Data JPA, and ☐ 検証. The '使用可能:' (Available) section has a search bar '検索する依存関係を入力' and a list of categories with expandable arrows: Alibaba, Amazon Web サービス, 開発ツール, Google Cloud Platform, I/O, メッセージング, Microsoft Azure, NoSQL, Observability, Ops, SQL, セキュリティ, Spring Cloud, Spring Cloud Circuit Breaker, and Spring Cloud Config. The '選択済み:' (Selected) section shows a list of dependencies with 'X' marks: Spring Boot DevTools, Spring Data JPA, H2 Database, Thymeleaf, and Spring Web. At the bottom right, there are buttons 'デフォルトにする' (Set Default) and '選択をクリア' (Clear Selection). The bottom bar contains a help icon, navigation buttons '< 戻る(B)' (Back), '次へ(N) >' (Next), '完了(F)' (Finish), and 'キャンセル' (Cancel).

ファイル構成



The screenshot displays the file explorer of an IDE for a project named 'spb_jpa_h2db1'. The project is a Spring Boot application. The main source code is located in 'src/main/java' under the package 'com.example.demo'. It includes a 'db' package with 'Staff.java', 'ServletInitializer.java', and 'SpbJpaH2db1Application.java'. Resources are in 'src/main/resources', including 'templates', 'static', and 'application.properties'. Test code is in 'src/test/java'. The project also includes a 'bin' directory, 'gradle' and 'src' folders, and build files: 'build.gradle', 'gradlew', 'gradlew.bat', 'HELP.md', and 'settings.gradle'.

- ▼ spb_jpa_h2db1 [boot] [devtools]
 - ▼ src/main/java
 - ▼ com.example.demo
 - ▼ db
 - > Staff.java
 - > ServletInitializer.java
 - > SpbJpaH2db1Application.java
 - ▼ src/main/resources
 - templates
 - static
 - application.properties
 - > src/test/java
 - > JRE システム・ライブラリー [JavaSE-11]
 - > プロジェクトと外部の依存関係
 - bin
 - > gradle
 - > src
 - build.gradle
 - gradlew
 - gradlew.bat
 - HELP.md
 - settings.gradle

データベースの設定(H2 database)

application.properties

spring.datasource.driverClassName: org.h2.Driver

spring.datasource.url: jdbc:h2:mem:test

spring.datasource.username: sa

spring.datasource.password:

spring.h2.console.enabled: true

#JPAによるテーブル生成の制御

spring.jpa.hibernate.ddl-auto=update

データベースの設定

schema.sql

data.sql

！ ポイント

schema.sql、data.sqlはバッティングしてしまうため作成しません
(実際には組み合わせて使う場合もあります)

エンティティの指定

Staff.java

```
import javax.persistence.Entity;  
import javax.persistence.Id;  
import javax.persistence.Table;
```

```
@Entity
```

```
@Table(name="staff")
```

エンティティと
テーブル名の指定

```
public class Staff {
```

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

主キーと
AUTOINCREMENT

```
    private Integer id;
```

```
    private String name;
```

```
    public Integer getId() {
```

```
        return id;
```

```
    }
```

```
    public void setId(Integer id) {
```

```
        this.id = id;
```

```
    }
```

```
    public String getName() {
```

```
        return name;
```

```
    }
```

```
    public void setName(String name) {
```

```
        this.name = name;
```

```
    }
```

```
}
```

実行後の確認

H2コンソールでの確認

The screenshot displays the H2 database console interface. The top toolbar includes icons for connection, schema, and execution, along with settings for 'Auto commit' (checked), 'Max rows' (1000), and 'Auto complete' (Off). The left sidebar shows the database structure for 'jdbc:h2:mem:test', with the 'STAFF' table highlighted by a red rectangle. The main area contains the SQL statement 'SELECT * FROM STAFF |' in the input field. Below the input field, the executed query 'SELECT * FROM STAFF;' is shown, followed by a table with two columns, 'ID' and 'NAME', and the message '(no rows, 5 ms)'. An 'Edit' button is located at the bottom of the result area.

jdbc:h2:mem:test

STAFF

INFORMATION_SCHEMA

Users

H2 1.4.200 (2019-10-14)

Run Run Selected Auto complete Clear SQL statement:

SELECT * FROM STAFF |

SELECT * FROM STAFF;

ID	NAME
----	------

(no rows, 5 ms)

Edit

A faint, light blue world map is visible in the background of the slide, centered behind the title text.

JPAの利用(MySQL)

データベース接続時に必要なもの

項目	目的
Spring Boot DevTools	開発を便利にするツール (コード変更時の自動再起動など)
Thymeleaf	テンプレートエンジン
Spring Web	Spring MVCを使う
Spring Data JPA	JPAライブラリを使う
MySQL Driver	データベースにMySQLを使う

新規 Spring スターター・プロジェクト依存関係

Spring Boot バージョン: 2.4.5

使用頻度高:

<input type="checkbox"/> H2 Database	<input type="checkbox"/> JDBC API	<input type="checkbox"/> Lombok
<input checked="" type="checkbox"/> MySQL Driver	<input checked="" type="checkbox"/> Spring Boot DevTools	<input checked="" type="checkbox"/> Spring Data JPA
<input checked="" type="checkbox"/> Spring Web	<input checked="" type="checkbox"/> Thymeleaf	<input type="checkbox"/> 検証

使用可能:

検索する依存関係を入力

- ▶ Alibaba
- ▶ Amazon Web サービス
- ▶ 開発ツール
- ▶ Google Cloud Platform
- ▶ I/O
- ▶ メッセージング
- ▶ Microsoft Azure
- ▶ NoSQL
- ▶ Observability
- ▶ Ops
- ▶ SQL
- ▶ セキュリティー
- ▶ Spring Cloud
- ▶ Spring Cloud Circuit Breaker
- ▶ Spring Cloud Config

選択済み:

- X Spring Boot DevTools
- X Spring Data JPA
- X MySQL Driver
- X Thymeleaf
- X Spring Web

デフォルトにする 選択をクリア

? < 戻る(B) 次へ(N) > 完了(F) キャンセル

ファイル構成

- ▼ spb_jpa_mysql1 [boot] [devtools]
 - ▼ src/main/java
 - ▼ com.example.demo
 - ▼ db
 - > Staff.java
 - > ServletInitializer.java
 - > SpbJpaMysql1Application.java
 - ▼ src/main/resources
 - templates
 - static
 - application.properties
 - > src/test/java
 - > JRE システム・ライブラリー [JavaSE-11]
 - > プロジェクトと外部の依存関係
 - bin
 - gradle
 - src
 - build.gradle
 - gradlew
 - gradlew.bat
 - HELP.md
 - settings.gradle

データベースの設定(MySQL)

application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/spb_db1?characterEncoding=UTF-8
```

```
spring.datasource.username=testuser
```

```
spring.datasource.password=testpass
```

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

```
#JPAによるテーブル生成の制御
```

```
spring.jpa.hibernate.ddl-auto=update
```

データベースの設定

schema.sql

data.sql

！ ポイント

schema.sql、data.sqlはバッティングしてしまうため作成しません
(実際には組み合わせて使う場合もあります)

エンティティの指定

Staff.java

```
import javax.persistence.Entity;  
import javax.persistence.Id;  
import javax.persistence.Table;
```

```
@Entity
```

```
@Table(name="staff")
```

エンティティと
テーブル名の指定

```
public class Staff {
```

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

主キーと
AUTOINCREMENT

```
    private Integer id;
```

```
    private String name;
```

```
    public Integer getId() {
```

```
        return id;
```

```
    }
```

```
    public void setId(Integer id) {
```

```
        this.id = id;
```

```
    }
```

```
    public String getName() {
```

```
        return name;
```

```
    }
```

```
    public void setName(String name) {
```

```
        this.name = name;
```

```
    }
```

```
}
```

実行後の確認

phpMyAdminでの確認

← サーバ: 127.0.0.1 > データベース: spb_db1

構造 SQL 検索 クエリ エクスポート インポート 操作 権限 ルーチン イベント

フィルタ

含まれている文字:

	テーブル ▲	操作	行	タイプ	照合順序	サイズ	オーバーヘッド
<input type="checkbox"/>	staff	★	0	InnoDB	utf8_general_ci	16.0 KiB	-
	1 テーブル	合計	0	InnoDB	utf8_general_ci	16.0 KiB	0 バイト

↑ ☐ すべてチェックする チェックしたものを:

印刷 データ辞書

テーブルを作成

名前: カラム数:

実行

A faint, light blue world map is visible in the background of the slide, centered behind the title text.

JPAによるDB操作 1 (H2 database)

JPAによるDB操作(SELECT)

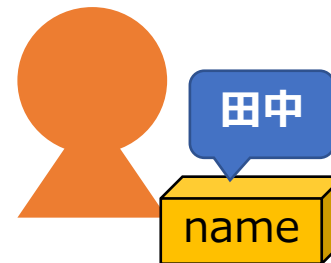
Webクライアント
(ブラウザ)



Controller



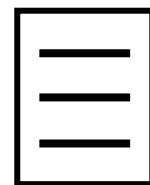
Model



データベース
(H2 database)



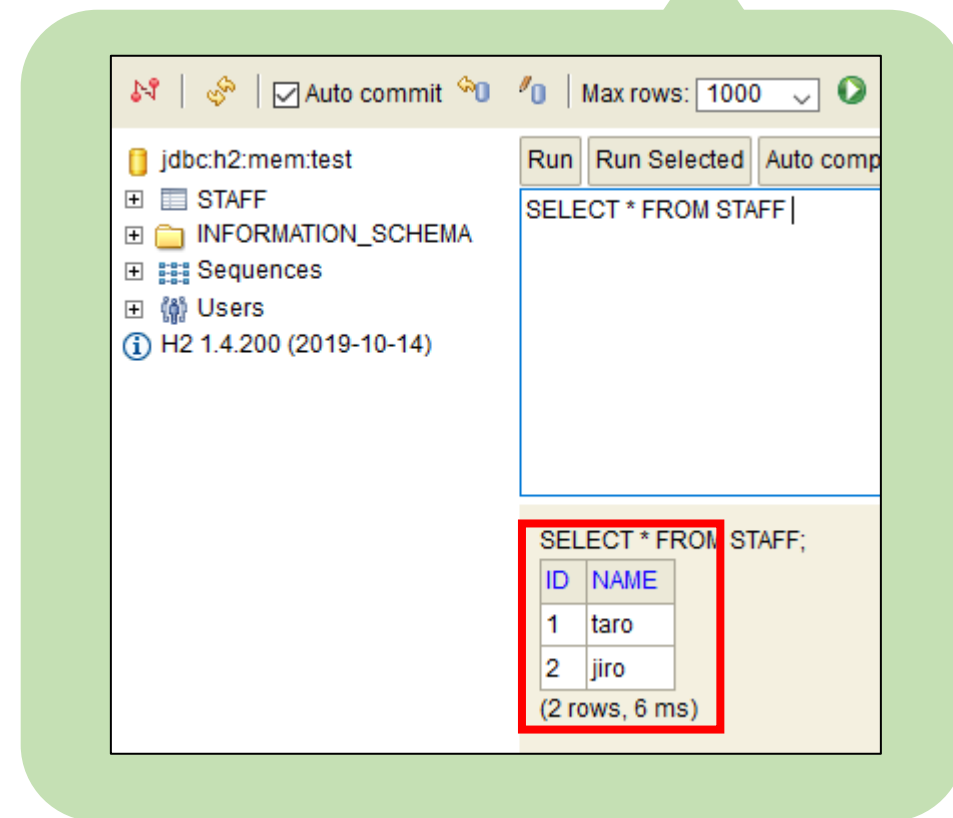
View



JPA(SELECT : 一覧表示)

1 taro
2 jiro

[H2コンソール](#)



アプリケーション作成時の選択

項目	目的
Spring Boot DevTools	開発を便利にするツール (コード変更時の自動再起動など)
Thymeleaf	テンプレートエンジン
Spring Web	Spring MVCを使う
Spring Data JPA	JPAライブラリを使う
H2 Database	データベースにH2を使う

 新規 Spring スターター・プロジェクト依存関係

Spring Boot バージョン: 2.4.5

使用頻度高:

☒ H2 Database

☐ JDBC API

☐ Lombok

☐ MySQL Driver

☒ Spring Boot DevTools

☒ Spring Data JPA

☒ Spring Web

☒ Thymeleaf

☐ 検証

使用可能:

検索する依存関係を入力

▶ Alibaba

▶ Amazon Web サービス

▶ 開発ツール

▶ Google Cloud Platform

▶ I/O

▶ メッセージング

▶ Microsoft Azure

▶ NoSQL

▶ Observability

▶ Ops

▶ SQL

▶ セキュリティ

▶ Spring Cloud

▶ Spring Cloud Circuit Breaker

▶ Spring Cloud Config

選択済み:

X Spring Boot DevTools

X Spring Data JPA

X H2 Database

X Thymeleaf

X Spring Web

デフォルトにする

選択をクリア

?

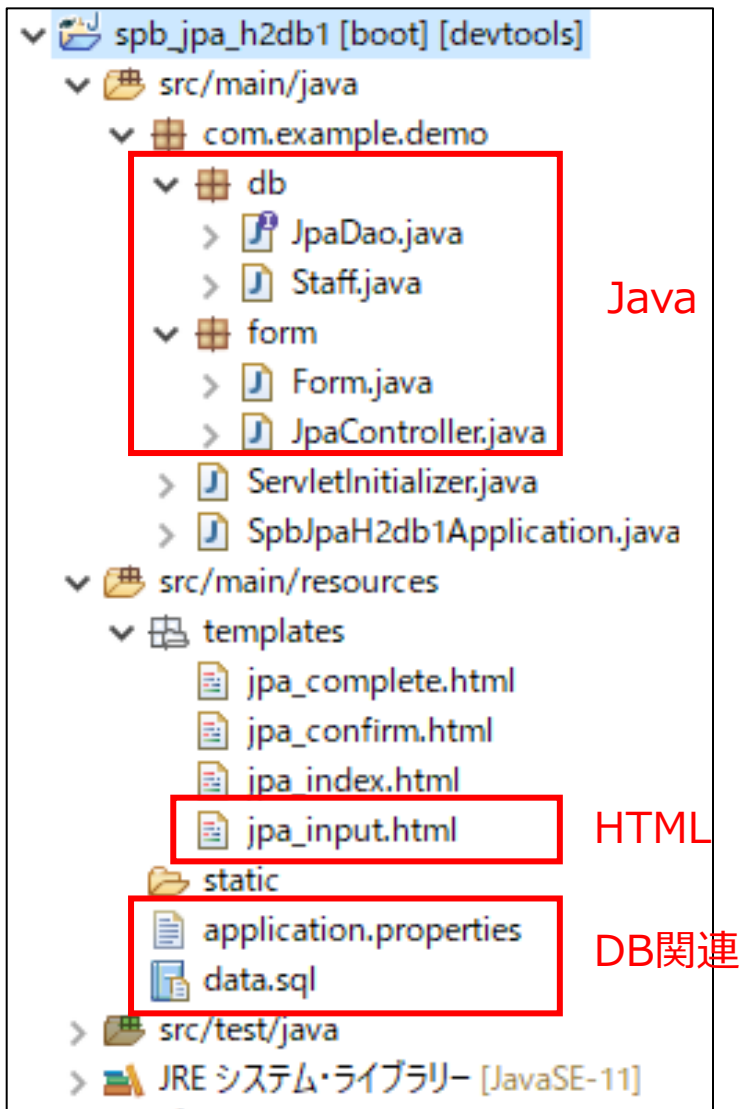
< 戻る(B)

次へ(N) >

完了(F)

キャンセル

ファイル構成



データベースの設定

application.properties

```
spring.datasource.driverClassName: org.h2.Driver  
spring.datasource.url: jdbc:h2:mem:test  
spring.datasource.username: sa  
spring.datasource.password:  
spring.h2.console.enabled: true
```

#JPAによるテーブル生成

```
spring.jpa.hibernate.ddl-auto=update
```

data.sql

```
INSERT INTO staff (id,name) VALUES (1,'taro');  
INSERT INTO staff (id,name) VALUES (2,'jiro');
```

※schema.sqlは作成しない

エンティティの指定

Staff.java(前半)

```
package com.example.demo.db;
```

```
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import javax.persistence.Table;
```

```
@Entity
```

```
@Table(name="staff")
```

```
public class Staff {
```

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
    private Integer id;
```

```
    private String name;
```

```
    public Staff() {}
```

```
    :
```

```
    :
```

※続きのコードは次のスライドにて

エンティティの指定

Staff.java(後半)

```
:
:
public Integer getId() {
    return id;
}
public void setId(Integer id) {
    this.id = id;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}

@Override
public String toString() {
    return "Staff [id=" + id + ", name=" + name + "];"
}
}
```

JpaController.java
内で使います

Model(DAO)の指定

JpaDao.java ※インターフェイスです

```
package com.example.demo.db;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface JpaDao extends JpaRepository<Staff, Long>{

}
```

Controllerの指定

JpaController.java

```
:
@Controller
public class JpaController {

    //DAOのオブジェクトが座る椅子を用意
    private final JpaDao jpadao;

    //DAOを予め控室に待機させておき、必要なときに呼んで座らせる
    @Autowired
    public JpaController(JpaDao jpadao) {
        this.jpadao = jpadao;
    }

    ※この部分は次のスライドにて記載

}
```

Controllerの指定

JpaController.java ※中の部分

```
:
// 【JPAでSELECT】
@RequestMapping("/")
public String index(Model model) {

    // JPAを用いたデータの取得
    List<Staff> stafflist=jpadao.findAll();

    // 取得できているかのチェック
    System.out.println(stafflist.get(0));

    // Viewに渡す
    model.addAttribute("stafflist", stafflist);

    // 【参考】上記を1行でまとめた形
    // model.addAttribute("stafflist", jpadao.findAll());

    return "jpa_index";
}
:
```

toStringメソッドの利用

Viewの指定

jpa_index.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>JPA</title>
<meta charset="utf-8" />
</head>
<body>
  <h1>JPA(SELECT : 一覧表示)</h1>
  <table>
    <tr th:each="staff : ${stafflist}">
      <td th:text="${staff.id}"></td>
      <td th:text="${staff.name}"></td>
    </tr>
  </table>
  <p><a href="http://localhost:8080/h2-console">H2コンソール</a></p>
</body>
</html>
```