

LU03c - Klassen und Tabellen synchronisieren

Ausgangslage

In der vorhergehenden Lektion haben Sie die Klasse `Book` aufgrund von der Tabelle `books` implementiert. Bei einer Änderung müssen Sie dadurch immer sowohl die Tabelle als auch die Klasse anpassen, da diese immer aufeinanderpassen müssen.

SQL books	Java Book					
<table border="1"> <tr> <td>publication_year</td> <td>INT</td> </tr> </table>	publication_year	INT	<table border="1"> <tr> <th>Book</th> </tr> <tr> <td> -id: int -title: String -author: String -isbn: String -publicationYear: int </td> </tr> <tr> <td> +Book(id: int, isbn: String, title: String, author: String, publicationYear: int) +getId(): int +getTitle(): String +setTitle(title: String): void +getAuthor(): String +setAuthor(author: String): void +getIsbn(): String +setIsbn(isbn: String): void +getPublicationYear(): int +setPublicationYear(publicationYear: int): void +toString(): String </td> </tr> </table>	Book	-id: int -title: String -author: String -isbn: String -publicationYear: int	+Book(id: int, isbn: String, title: String, author: String, publicationYear: int) +getId(): int +getTitle(): String +setTitle(title: String): void +getAuthor(): String +setAuthor(author: String): void +getIsbn(): String +setIsbn(isbn: String): void +getPublicationYear(): int +setPublicationYear(publicationYear: int): void +toString(): String
publication_year	INT					
Book						
-id: int -title: String -author: String -isbn: String -publicationYear: int						
+Book(id: int, isbn: String, title: String, author: String, publicationYear: int) +getId(): int +getTitle(): String +setTitle(title: String): void +getAuthor(): String +setAuthor(author: String): void +getIsbn(): String +setIsbn(isbn: String): void +getPublicationYear(): int +setPublicationYear(publicationYear: int): void +toString(): String						

Um unnötige Arbeitsschritte zu vermeiden, können wir die eine Seite aus der anderen generieren:

- Java-first: Aus den Java-Klassen werden die entsprechenden SQL-Tabellen generiert.
- SQL-first: Aus den SQL-Tabellen werden die entsprechende Java-Klassen generiert.

JPA/Hibernate

Wir wählen den Java-first-Ansatz mit JPA/Hibernate.

Die `build.gradle` wurde im Commit `TODO` bereits mit den passenden Abhängigkeiten ergänzt. Ebenfalls existiert eine Konfiguration `persistence.xml`. Gewisse Einstellungen wie `jakarta.persistence.jdbc.url`, `jakarta.persistence.jdbc.user` oder `jakarta.persistence.jdbc.password` werden nicht mit der Konfigurationsdatei `committed`, sondern können wie bis anhin in `config.properties` geschrieben und ersetzen die bisherigen Variablen.

Dank Hibernate können die Eigenschaften der Tabelle als Annotationen (`@...`) in der Java-Klasse gesetzt werden. Ein leerer Standard-Constructor ist nun auch nötig, damit Hibernate im Hintergrund eine neue Instanz erstellen kann. Dadurch sieht die Klasse `Book` neu in etwa so aus.

```
// package ...
```

```
import jakarta.persistence.*;

@Entity
@Table(name = "books")
public class Book {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @Column(name = "isbn", nullable = false, length = 20, unique = true)
    private String isbn;

    @Column(name = "title", nullable = false, length = 255)
    private String title;

    @Column(name = "author", nullable = false, length = 255)
    private String author;

    @Column(name = "publication_year")
    private Integer publicationYear;

    public Book(){}

    public Book(Integer id, String isbn, String title, String author,
Integer publicationYear) {
        this.id = id;
        this.isbn = isbn;
        this.title = title;
        this.author = author;
        this.publicationYear = publicationYear;
    }

    // toString, equals(), hashCode() if needed
    ...

    // Getter & Setter
    ...
}
```

From:
<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:
<https://wiki.bzz.ch/de/modul/ffit/3-jahr/java/learningunits/lu03/c?rev=1756742925>

Last update: **2025/09/01 18:08**

