

# Datenbankmigrationen mit Flyway in Spring Boot

Hier ist eine prägnante, praktische Anleitung zur Verwendung von Flyway mit einer Spring Boot-Anwendung zur Verwaltung Ihres Datenbankschemas:

---

## Was ist Flyway?

Flyway ist ein Datenbankmigrationswerkzeug, das Ihnen hilft, Ihr Datenbankschema unter Versionskontrolle zu stellen. Es wendet SQL-Skripte (Migrations) in einer bestimmten Reihenfolge an, um Ihre Datenbank weiterzuentwickeln, was es ideal für Spring Boot-Projekte macht, bei denen Konsistenz und Automatisierung entscheidend sind.

---

## Schritt 1: Flyway-Abhängigkeit hinzufügen

Um loszulegen, fügen Sie Flyway zu Ihrem Projekt hinzu. Wenn Sie **Maven** verwenden, aktualisieren Sie Ihre `pom.xml`:

```
<dependency>
  <groupId>org.flywaydb</groupId>
  <artifactId>flyway-core</artifactId>
</dependency>
```

Für **Gradle** fügen Sie dies zu Ihrer `build.gradle` hinzu:

```
implementation 'org.flywaydb:flyway-core'
```

Dies zieht Flyway in Ihre Spring Boot-Anwendung ein.

---

## Schritt 2: Datenbankverbindung konfigurieren

Flyway muss wissen, welche Datenbank es verwalten soll. Definieren Sie Ihre Datenbankverbindung in `application.properties` (oder `application.yml`). Zum Beispiel mit PostgreSQL:

```
spring.datasource.url=jdbc:postgresql://localhost:5432/mydb
spring.datasource.username=myuser
spring.datasource.password=mypassword
```

Spring Boot konfiguriert Flyway automatisch so, dass es diese Datenquelle verwendet, sodass hier keine zusätzliche Einrichtung erforderlich ist.

---

### Schritt 3: Migrationsskripte erstellen

Flyway verwendet versionierte SQL-Skripte, um Ihre Datenbank zu aktualisieren. Platzieren Sie diese Skripte im Standardverzeichnis: `src/main/resources/db/migration`. Jedes Skript muss diesem Namensmuster folgen: `V<version>__<beschreibung>.sql`. Die Versionsnummer stellt sicher, dass die Migrations in der richtigen Reihenfolge ausgeführt werden.

Erstellen Sie beispielsweise eine Datei mit dem Namen `V1__create_users_table.sql`:

```
CREATE TABLE users (  
    id SERIAL PRIMARY KEY,  
    username VARCHAR(255) NOT NULL,  
    email VARCHAR(255) NOT NULL  
);
```

Speichern Sie sie in `src/main/resources/db/migration`.

---

### Schritt 4: Ihre Anwendung starten

Starten Sie Ihre Spring Boot-Anwendung (z.B. über `mvn spring-boot:run` oder Ihre IDE). Flyway wird: - Die Datenbank auf eine Migrationshistorie überprüfen. - Alle ausstehenden Migrations (wie `V1__create_users_table.sql`) der Reihe nach anwenden.

Dies geschieht automatisch beim Starten dank der Integration von Spring Boot –keine manuelle Eingriffe erforderlich.

---

### Schritt 5: Flyway anpassen (optional)

Sie können das Verhalten von Flyway mit Eigenschaften in `application.properties` anpassen. Hier sind einige gängige Optionen:

```
spring.flyway.locations=classpath:db/migration # Wo die Skripte gefunden werden  
spring.flyway.baseline-on-migrate=true         # Existierende Datenbanken verwalten  
spring.flyway.enabled=true                     # Flyway ein-/ausschalten
```

Diese ermöglichen es Ihnen, Flyway an die Bedürfnisse Ihres Projekts anzupassen.

---

## Fortgeschrittene Funktionen

Flyway bietet mehr als nur grundlegende Migrations: - **Java-basierte Migrations**: Schreiben Sie Migrations in Java für komplexe Logik. - **Wiederholbare Migrations**: Verwenden Sie `R__<beschreibung>.sql` für Skripte, die bei Änderungen erneut ausgeführt werden (z.B. Ansichten). - **Migrations rückgängig machen**: Änderungen rückgängig machen (erfordert Flyway Pro).

Besuchen Sie die Flyway-Dokumentation für eine tiefere Erforschung.

---

## Zusammenfassung

Um Flyway in Spring Boot zu verwenden: 1. Fügen Sie die Flyway-Abhängigkeit zu Ihrem Projekt hinzu. 2. Richten Sie Ihre Datenbankverbindung in `application.properties` ein. 3. Schreiben Sie versionierte SQL-Skripte in `src/main/resources/db/migration`. 4. Starten Sie Ihre App –Flyway wendet die Migrations automatisch an.

Flyway vereinfacht die Verwaltung des Datenbankschemas mit Versionskontrolle und Automatisierung und ergänzt perfekt die gestraffte Entwicklungserfahrung von Spring Boot.