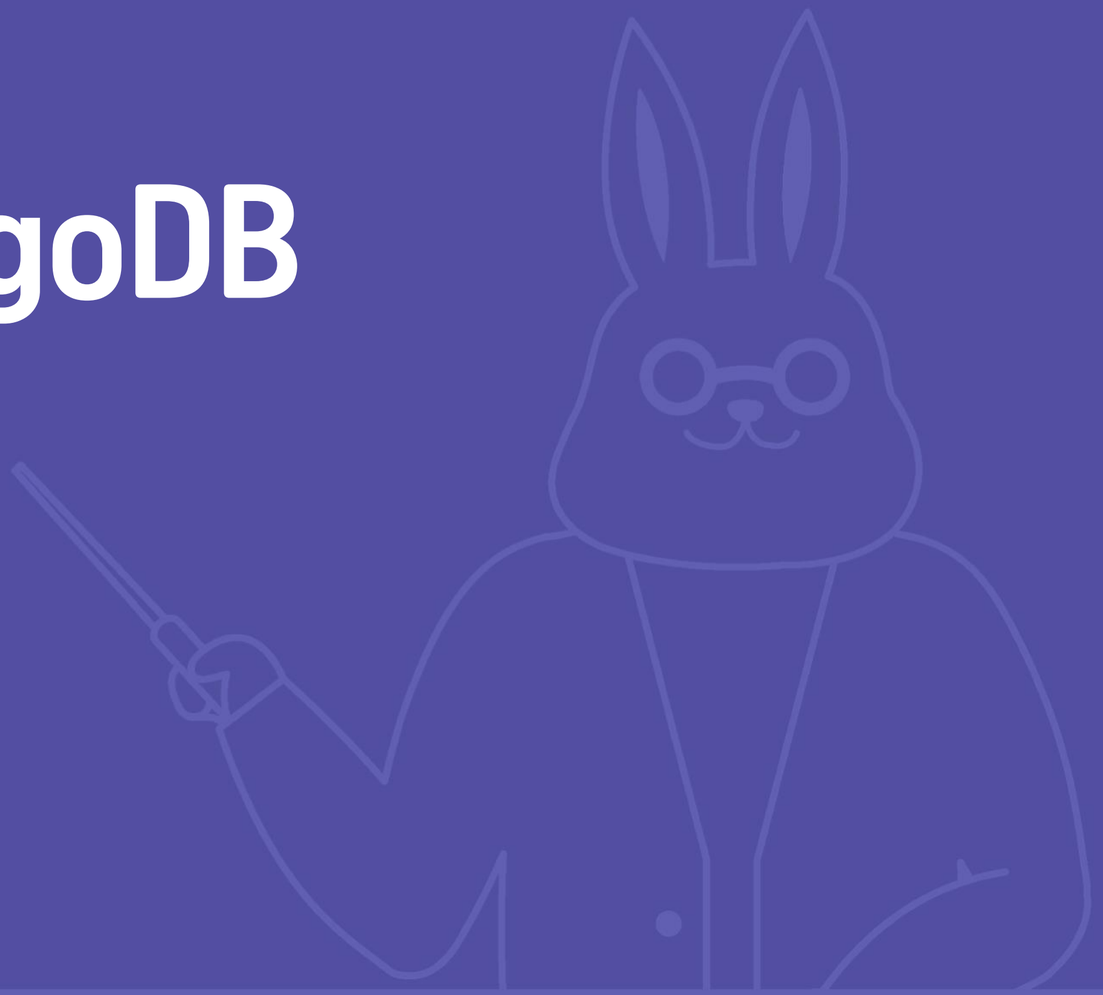
Express.js와 MongoDB

04 Mongoose 활용하기



/* elice */

01

Express.js + Mongoose ODM

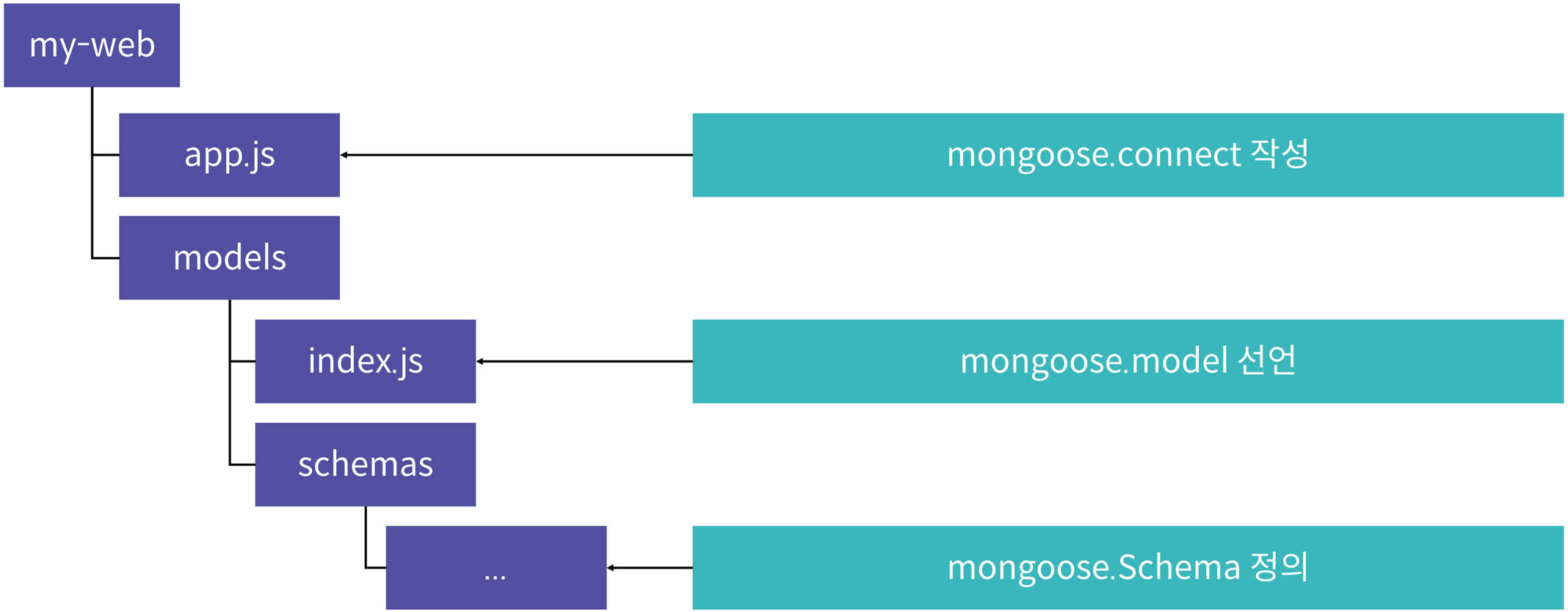
Express.js에서 Mongoose ODM 사용하기

Express.js는 **프로젝트 구조를 자유롭게 구성**할 수 있기 때문에
어느 부분에 Mongoose ODM을 위치시키면 좋을지
적절한 위치를 결정하는 것이 중요

Mongoose ODM 위치 정하기

일반적으로 **models** 디렉터리에 **Schema와 Model을 같이** 위치
app 객체는 어플리케이션 시작을 의미하는 부분이므로
해당 부분에 데이터베이스 연결을 명시하는 mongoose.connect를 위치

Mongoose ODM 위치 정하기



Mongoose ODM 커넥션 이벤트

Express.js 어플리케이션은 **종료되지 않고 동작**하기 때문에
계속해서 **데이터베이스가 정상적으로 동작하는지**를 파악하기 위해
동작 중에 발생하는 **데이터베이스 연결 관련 이벤트에 대한 처리**를 하는 것이 좋음

Mongoose ODM 커넥션 이벤트

connection events

```
mongoose.connect('----');

mongoose.connection.on('connected', () => {
});

mongoose.connection.on('disconnected', () => {
});

mongoose.connection.on('reconnected', () => {
});

mongoose.connection.on('reconnectFailed', () => {
});
```

connected - 연결 완료

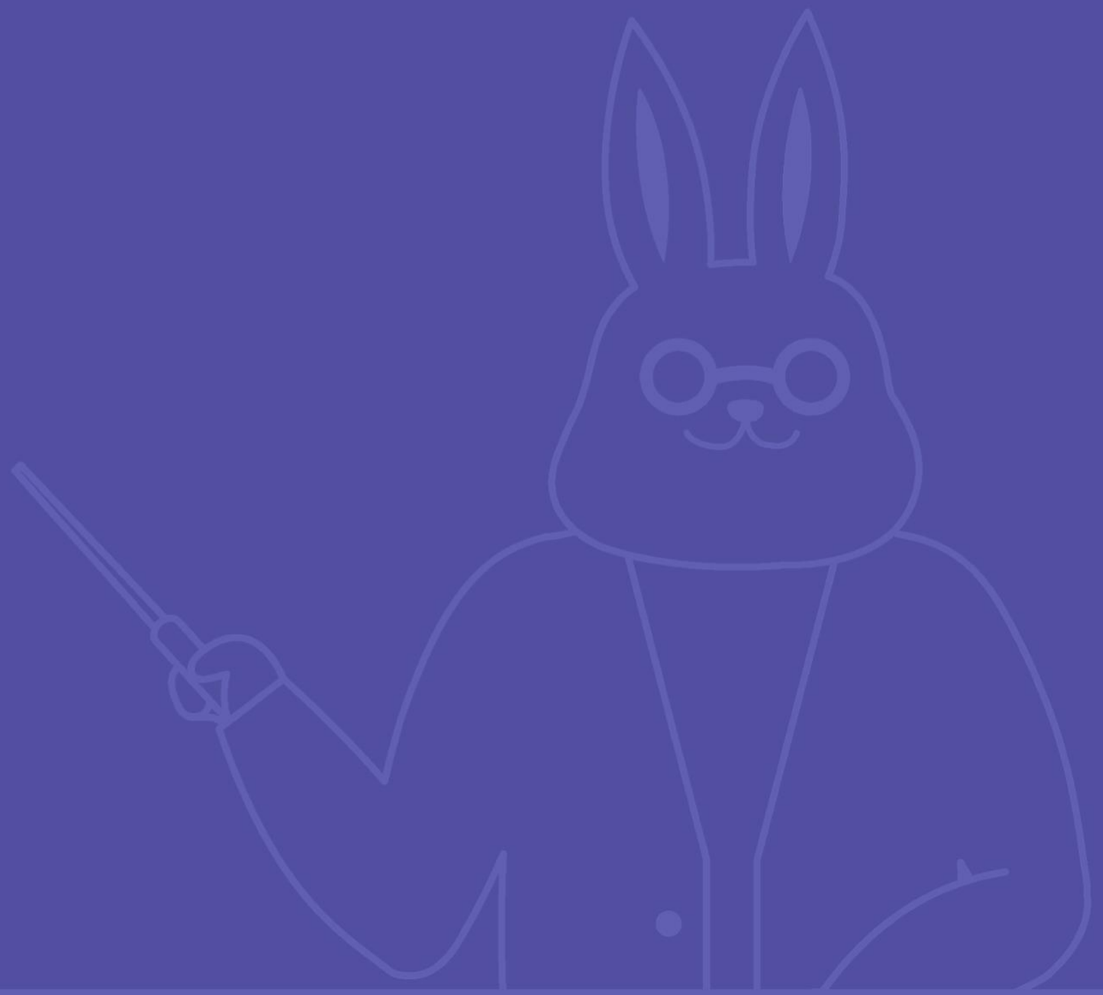
disconnected - 연결이 끊김

reconnected - 재연결 완료

reconnectFailed - 재연결 시도 횟수 초과

02

Sequelize ORM



Sequelize ORM 이란?

Object-**R**elational **M**apping

MySQL, PostgreSQL 등의 **RDBMS**를 이용하는 간단한 방법
ODM이 단순히 **모델에 집중**하여 관리하는 것에 반해,
ORM은 **테이블 관계**와 **쿼리** 등의 기능을 **더욱 단순화**하는 용도로 주로 사용

Sequelize ORM 사용하기

본 강의는 **MongoDB에 집중**하기 위해
RDB와 ORM에 대한 **자세한 내용은 다루지 않음**
ORM의 **작성**과 **사용 방법**의 **예제코드**만 함께 살펴봄

디비 연결

connect db

```
const sequelize = new Sequelize('database', 'username', 'password', {
  host: 'localhost',
  dialect: 'mysql'
});
```

sequelize도 **연결을 관리**하는 간단한 방법을 제공
mongoose가 MongoDB만 연결이 가능한 데에 반해
sequelize는 MySQL, PostgreSQL, SQLite 등 **다양한 RDBMS 에 연결 가능**

스키마 작성

define

```
const User = sequelize.define('User', {
  name: {
    type: DataTypes.STRING(10),
    allowNull: false
  },
  age: {
    type: DataTypes.Integer,
  }
}, {});
```

sequelize는 **define**을 통해
Schema를 생성

mongoose.Schema와 유사하지만

sequelize는 Schema가 **DDL도 생성**해 줌

스키마 작성 - 관계 정의

relation

```
User.hasMany(Post);
Post.belongsTo(User);
Foo.belongsToMany(Bar);
Bar.belongsToMany(Foo);
```

sequelize를 이용하면 **테이블 간의 관계**를 **Code-Level로 관리**할 수 있음

이를 이용하면 **외래키 설정과 제약조건**까지 **DDL로 생성**해 줌

또한 **다대다 관계 설정**을 통해 **relation table**도 자동으로 생성해 줌

쿼리

query

```
User.findAll({
  where: {
    name: 'elice',
    age: {
      [Op.lt]: 20,
      [Op.gte]: 10,
    },
  },
  include: User,
});
```

Operator를 이용해
SQL 쿼리를 코드로 작성 가능

스키마의 관계 설정을 한 경우, **include**를
사용하여 **자동으로 join 쿼리 생성** 가능

Synchronization

sync

```
sequelize.sync();
```

define된 model 데이터를 바탕으로 **DDL을 자동으로 실행**해 줌
-> 직접 데이터베이스에 접속하여 **테이블 생성 및 관리를 할 필요가 없음**
-> 자동으로 생성된 DDL을 따르지 않으면 **테이블 관리가 어려워 짐**

Sequelize ORM 정리

Sequelize ORM 을 사용하면 데이터베이스에 **직접 DDL을 하지 않고 JavaScript 코드로** 테이블 및 관계를 관리할 수 있음
또한 RDB의 어려운 점 중 하나인 **join을 간단하게 사용**할 수 있음

연락처

TEL
070-4633-2015

WEB
https://elice.io

E-MAIL
contact@elice.io