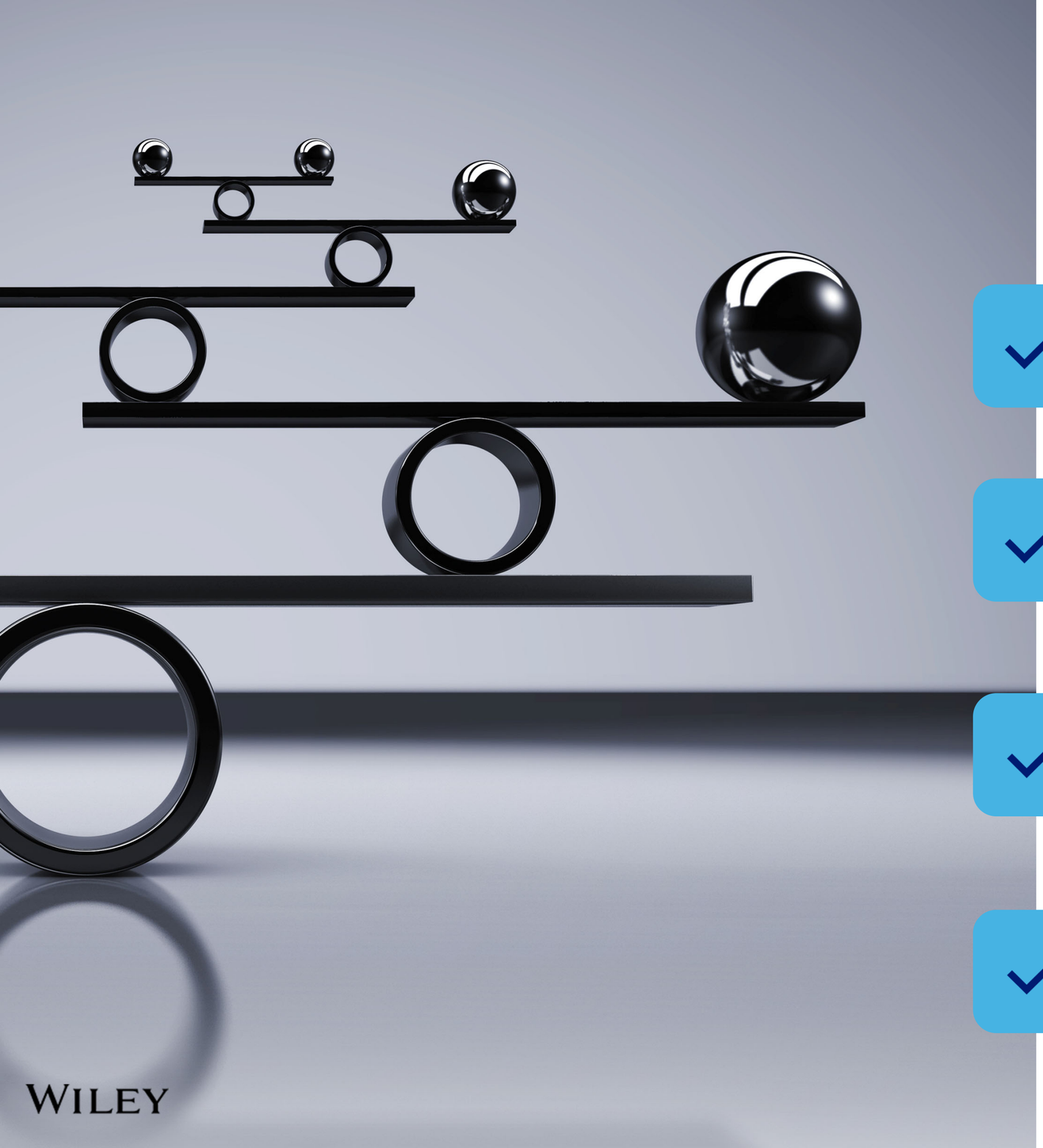
WILEY

ENABLING DISCOVERY | POWERING EDUCATION | SHAPING WORKFORCES

Integrate Node with MongoDB

Connecting with Database



Objectives

- Connecting Node with MongoDB
- MongoDB NodeJS Driver
- Understanding Mongoose
- Working Object Relation Mapping

Objectives

- Working with Mongoose Connections
- Working with Schema and its types
- Working with Mongoose Models

Connecting Express with Mongodb

Mongoose Introduction

- Mongoose is one of the Node.js libraries that provides MongoDB object mapping.
- Mongoose is a library of Node.js, it provides interaction with MongoDB using ORM.
- Mongoose provides a straight-forward, schema-based solution to model our application data.
- It includes built-in type casting, validation, query building, business logic hooks and more.

Defining your schema

- **Everything in Mongoose starts with a Schema.**
- **Each schema maps to a MongoDB collection and defines the shape of the documents.**

```
import mongoose from 'mongoose';
const { Schema } = mongoose;

const blogSchema = new Schema({
  title:  String, // String is shorthand for {type: String}
  author: String,
  body:   String,
  comments: [{ body: String, date: Date }],
  date: { type: Date, default: Date.now },
  hidden: Boolean,
  meta: {
    votes: Number,
    favs:  Number
  }
});
```

WHAT IS A SCHEMA TYPE?

- Mongoose schema is the configuration object for a Mongoose model.
- A Schema Type is then a configuration object for an individual property.
- A Schema Type says what type a given path should have, whether it has any getters/setters, and what values are valid for that path.

```
const schema = new Schema({ name: String });
schema.path('name') instanceof mongoose.SchemaType; // true
schema.path('name') instanceof mongoose.Schema.Types.String; // true
schema.path('name').instance; // 'String'
```

Mongoose Connection

- You can connect to MongoDB with the mongoose.connect( ) method.

```
mongoose.connect('mongodb://localhost:27017/myapp', {useNewUrlParser: true});
```

- This is the minimum needed to connect the myapp database running locally on the default port (27017). If connecting fails on your machine, try using 127.0.0.1 instead of localhost.

Mongoose Models

- Models are fancy constructors compiled from Schema definitions.
- An instance of a model is called a document.
- Models are responsible for creating and reading documents from the MongoDB database.
- When you call mongoose.model() on a schema, Mongoose compiles a model for you.

```
const schema = new mongoose.Schema({ name: 'string', size: 'string' });
const Tank = mongoose.model('Tank', schema);
```

- Mongoose automatically looks for the plural, lowercased version of your model's name.
- Thus, for the example above, the model Tank is for the tanks collection in the database.

MONGODB NODEJS DRIVER

- The MongoDB Node.js driver allows Node.js applications to connect to MongoDB and work with data.
- The driver features an asynchronous API which allows you to interact with MongoDB.
- This driver uses Promises or via traditional call-backs while connecting with database.
- Given that you have created your own project using `npm init` we install the mongodb driver and it's dependencies by executing the following `NPM` command. This will download the MongoDB driver and add a dependency entry in your `package.json` file.

```
npm install mongodb --save
```

Thank you!