

全部课程 (/courses/) / Node.js使用Sequelize操作MySQL (/courses/509) / Node.js使用Sequelize操作MySQL

在线实验，请到PC端体验

一、实验介绍

1.1 实验内容

Sequelize 是一个 Node.js 平台基于 Promise 的ORM。用于操作管理 MySQL、Postgres、SQLite 等关系型数据库。本课程主要学习使用 Sequelize 操作 MySQL 数据库。Sequelize 官方文档：<http://docs.sequelizejs.com/en/latest/> (<http://docs.sequelizejs.com/en/latest/>)

1.2 实验知识点

- Sequelize 的使用

1.3 实验环境

- Node.js 6.x

1.4 适合人群

本课程难度为一般，属于初级级别课程，适合具有 Node.js 基础的用户学习 Node.js 连接 MySQL 数据库的相关开发。

二、开发准备

2.1 初始化项目

首先，创建一个 Node.js 项目：

```
$ mkdir myblog
$ cd myblog
myblog/ $ npm init
```

npm init 命令会提示输入一系列项目信息，直接一路回车即可。

然后安装 Sequelize 和 MySQL 包：

```
myblog/ $ npm install sequelize --save
myblog/ $ npm install mysql --save
```

OK，安装完毕！

三、实验步骤

3.1 连接数据库

首先，需要启动 MySQL 数据库：

```
$ sudo service mysql start
```

因为 Sequelize 不能创建数据库，所以需要手动创建一个数据库：

```
$ mysql -uroot
> create database blog;
>
```

动手实践是学习 IT 技术最有效的方式！

开始实验

按 Ctrl+c 键退出MySQL命令行界面。

创建了数据库之后，就可以通过 Sequelize 连接到数据库了，在 myblog 文件夹中新建 db.js 文件：

```
// db.js

var Sequelize = require('sequelize');

/*
// new Sequelize(database, [username=null], [password=null], [options={}]
// class Sequelize 接收4个参数，后三个参数是可选的

// 没有密码和options
var sequelize = new Sequelize('database', 'username')

// 没有options
var sequelize = new Sequelize('database', 'username', 'password')

// 没有密码有options
var sequelize = new Sequelize('database', 'username', null, {})

// 都有
var sequelize = new Sequelize('my_database', 'john', 'doe', {})

// new Sequelize(uri, [options={}]
// 通过uri连接数据库
var sequelize = new Sequelize('mysql://localhost:3306/database', {})
*/

module.exports = new Sequelize('blog', 'root', null, {
  host: 'localhost', // 数据库地址
  dialect: 'mysql', // 指定连接的数据库类型
  pool: {
    max: 5, // 连接池中最大连接数量
    min: 0, // 连接池中最小连接数量
    idle: 10000 // 如果一个线程 10 秒钟内没有被使用过的话，那么就释放线程
  }
});
```

连接数据库初始化时，Sequelize 会设置一个连接池，所以每个数据库创建一个实例即可。

连接池是创建和管理一个连接的缓冲池的技术，这些连接准备好被任何需要它们的线程使用。

3.2 创建用户表

下面来创建一张表模型（model），在 myblog 文件夹中新建 user.js 文件：

```

// user.js

var Sequelize = require('sequelize');
var sequelize = require('./db');

// 创建 model
var User = sequelize.define('user', {
  userName: {
    type: Sequelize.STRING, // 指定值的类型
    field: 'user_name' // 指定存储在表中的键名称
  },
  // 没有指定 field, 表中键名称则与对象键名相同, 为 email
  email: {
    type: Sequelize.STRING
  }
}, {
  // 如果为 true 则表的名称和 model 相同, 即 user
  // 为 false MySQL创建的表名称会是复数 users
  // 如果指定的表名称本就是复数形式则不变
  freezeTableName: false
});

// 创建表
// User.sync() 会创建表并且返回一个Promise对象
// 如果 force = true 则会把存在的表 (如果users表已存在) 先销毁再创建表
// 默认情况下 force = false
var user = User.sync({ force: false });

// 添加新用户
exports.addUser = function(userName, email) {
  // 向 user 表中插入数据
  return User.create({
    user_name: userName,
    email: email
  });
};

// 通过用户名查找用户
exports.findByName = function(userName) {
  return User.findOne({ where: { user_name: userName } });
};

```

3.3 测试用户表

在 myblog 文件夹中创建 testUser.js 文件, 测试user.js:

```

// testUser.js

var user = require('./user');

// 添加用户
user.addUser('jack', 'jack@163.com').then(function() {
  // 查询新添加的用户
  return user.findByName('jack');
}).then(function(user) {
  console.log('*****');
  console.log('user name: ', user.user_name);
  console.log('user email: ', user.email);
});

```

运行 testUser.js 文件:

```
myblog/ $ node testUser.js
```

可以看到终端的最后打印出了用户名和用户邮箱, 说明创建表和插入数据成功了。也可以登录到MySQL, 手动查询是否成功创建了用户表和插入了数据。

3.4 创建文章表

在 myblog 文件夹中创建 post.js 文件:

动手实践是学习 IT 技术最有效的方式!

开始实验

```
// post.js

var Sequelize = require('sequelize');
var sequelize = require('./db');

var Post = sequelize.define('post', {
  // 文章标题
  title: {
    type: Sequelize.STRING
  },
  // 文章内容
  content: {
    type: Sequelize.STRING
  },
  // 文章发表时间
  create_at: {
    type: Sequelize.DATE
  }
}, {
  freezeTableName: false
});

var post = Post.sync();

// 发表新文章
exports.newPost = function(title, content) {
  return post.then(function() {
    Post.create({
      title: title,
      content: content,
      create_at: Date.now()
    });
  });
};

// 查找所有文章
exports.findAllPosts = function() {
  return Post.findAll();
};

// 通过 ID 查找文章
exports.findById = function(id) {
  return Post.findById(id);
};
```

3.5 测试文章表

新建 testPost.js 文件:

```
var post = require('./post');

// 插入一篇文章
post.newPost('post title', 'post content').then(function() {
  // 通过ID查找文章
  return post.findById(1);
}).then(function(p) {
  console.log('*****');
  console.log('post title: ', p.title);
  console.log('post content: ', p.content);
});
```

运行 testPost.js 文件:

```
myblog/ $ node testPost.js
```

可以看到终端的最后打印出了文章标题文章内容, 说明创建表和插入数据成功了。也可以登录到 MySQL, 手动查询是否成功创建了文章表和插入了数据。

通过 Sequelize 操作 MySQL 数据库很简单吧!

更多功能可以看看Sequelize文档: <http://docs.sequelizejs.com/en/latest/> (<http://docs.sequelizejs.com/en/latest/>)

四、总结

动手实践是学习 IT 技术最有效的方式!

开始实验

Sequelize 是一个 Node.js 平台基于 Promise 的ORM。用于操作管理 MySQL、Postgres、SQLite 等关系型数据库。本课程主要学习使用 Sequelize 操作 MySQL 数据库。Sequelize 官方文档：<http://docs.sequelizejs.com/en/latest/> (<http://docs.sequelizejs.com/en/latest/>)

课程教师



forever

共发布过8门课程

[查看老师的所有课程 > \(/teacher/45\)](/teacher/45)

前置课程

[Node.js 教程 \(/courses/44\)](/courses/44)

进阶课程

[Node.js 经典项目实战 \(/courses/455\)](/courses/455)



动手做实验，轻松学IT



公司

<http://weibo.com/shiyanlou2013>

[关于我们 \(/aboutus\)](/aboutus)

[联系我们 \(/contact\)](/contact)

[加入我们 \(http://www.simplecloud.cn/jobs.html\)](http://www.simplecloud.cn/jobs.html)

[技术博客 \(https://blog.shiyanlou.com\)](https://blog.shiyanlou.com)

服务

[企业版 \(/saas\)](/saas)

[实战训练营 \(/bootcamp/\)](/bootcamp/)

[会员服务 \(/vip\)](/vip)

[实验报告 \(/courses/reports\)](/courses/reports)

[常见问题 \(/questions/?\)](/questions/)

<tag=%E5%B8%B8%E8%A7%81%E9%97%AE%E9%A2%98>

[隐私条款 \(/privacy\)](/privacy)

合作

[我要投稿 \(/contribute\)](/contribute)

[教师合作 \(/labs\)](/labs)

[高校合作 \(/edu/\)](/edu/)

[友情链接 \(/friends\)](/friends)

[开发者 \(/developer\)](/developer)

学习路径

[Python学习路径 \(/paths/python\)](/paths/python)

[Linux学习路径 \(/paths/linuxdev\)](/paths/linuxdev)

[大数据学习路径 \(/paths/bigdata\)](/paths/bigdata)

[Java学习路径 \(/paths/java\)](/paths/java)

[PHP学习路径 \(/paths/php\)](/paths/php)

[全部 \(/paths/\)](/paths/)

Copyright ©2013-2017 实验楼在线教育 | 蜀ICP备13019762号 (<http://www.miibeian.gov.cn/>)

动手实践是学习 IT 技术最有效的方式!

[开始实验](#)