### 搭建node+mysql+vue框架步骤

1. **全局安装express-generator**

npm install express-generator -g

1. **全局安装express**

npm install express -g

**3、使用express构建/初始化项目**

Express -e [name]

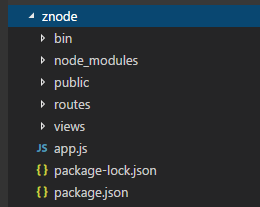
**4、express常用命令**

用法: express [options] [dir]

可用Options参数:

|  |  |  |
| --- | --- | --- |
| --version |  | output the version number |
| -e | --ejs | add ejs engine support |
| --pug |  | add pug engine support |
| --hbs |  | add handlebars engine support |
| -H | --hogan | add hogan.js engine support |
| -v | --view <engine> | add view <engine> support  (dust|ejs|hbs|hjs|jade|pug|twig|vash)  (defaults to jade) |
| -c | --css <engine> | add stylesheet <engine> support  (less|stylus|compass|sass)  (defaults to plain css) |
| --git |  | add .gitignore |
| -f | --force | force on non-empty directory |
| -h | --help | output usage information |

**5、项目目录说明**



bin: 用于应用启动的配置文件

node\_modules： 存放项目的依赖库

public: 静态资源目录

routes：可以认为是controller（控制器）目录

views: ejs模板目录，可以认为是view(视图)目录

app.js 程序main文件，入口文件

**6、进入项目目录，使用安装依赖**

npm install

**7、运行项目：**

npm start 或者 PORT=5000 npm start

**8、全局安装supervisor**

supervisor：npm install supervisor -g

修改package scripts中start设置，改为 “supervisor ./bin/www”

supervisor是一个node模块，用来启动node项目。

supervisor可以监控一些文件，当这些文件发生变化时自动刷新程序，不用重新启动node程序。

当程序崩溃时，supervisor会重新启动程序。

**9、安装config-lite模块配置参数**

config-lite 是一个轻量的读取配置文件的模块。

config-lite 会根据环境变量（NODE\_ENV）的不同从当前执行进程目录下的 config 目录加载不同的配置文件。

如果不设置 NODE\_ENV，则读取默认的 default 配置文件，如果设置了 NODE\_ENV，则会合并指定的配置文件和 default 配置文件作为配置，config-lite 支持 .js、.json、.node、.yml、.yaml 后缀的文件。

例：

如果程序以 NODE\_ENV=test node app 启动，则通过 require('config-lite') 会依次降级查找 config/test.js、config/test.json、config/test.node、config/test.yml、config/test.yaml 并合并 default 配置;

如果程序以 NODE\_ENV=production node app 启动，则通过 require('config-lite') 会依次降级查找 config/production.js、config/production.json、config/production.node、config/production.yml、config/production.yaml 并合并 default 配置。

补充：

环境变量在 Node.js 中可以通过process.env来访问当前的环境变量信息：

设置环境变量：

环境变量的名字一般为大写，多个单词之间可通过下划线来连接。

Windows 系统下可通过set命令来设置环境变量，比如：$ set HELLO\_MSG="Hello, world!"

Linux 系统下可通过export命令来设置，比如：$ export HELLO\_MSG="Hello, world!"

在 Node.js 中读取环境变量：

创建文件1.js，代码如下：console.log( process.env.HELLO\_MSG );

然后在命令行中执行：$ export HELLO\_MSG="Hello, world" && node 1.js

在项目根目录创建config文件夹，并在文件夹下创建default.js，在default.js中添加基本配置，

可以配置名称、端口等基本信息

使用方法：

var config = require(‘config-lite)(\_\_dirname);

//使用config.name ....

**10、创建数据库**

1）登录数据库： mysql -uroot -p

2）输入密码登录进去

3）创建项目新的数据库：CREATE DATABASE IF NOT EXISTS [name] CHARACTER SET UTF8;

4）查看所有的数据库：SHOW DATABASES;

5）为新建数据库设置权限、用户、登录密码：

GRANT ALL ON [name].\* to zyj@'%' IDENTIFIED BY '123'; 或

GRANT ALL ON [name].\* to zyj@localhost IDENTIFIED BY '456';

第一句表示zyj用户可以使用123密码从任何地址连接[name]数据库，具有操作数据中所有表的权限

第二句表示zyj用户可以使用456密码从localhost谅解[name]数据库，具有操作数据库中所有表的权限

6）退出mysql重新登录

EXIT;

mysql -uzyj -p

输入刚才设置的密码登录进去，使用 SHOW DATABASES; 查看当前数据库

7）为项目数据库添加user表格及用户数据

USE [NAME]; //进入到当前数据库

SELECT DATABASE(); //查看当前正在使用的数据库

CREATE TABLE user( //创建user表格

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(16) NOT NULL,

create\_date TIMESTAMP NULL DEFAULT now()

)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE UNIQUE INDEX user\_id on user(name); //表上创建一个唯一的索引

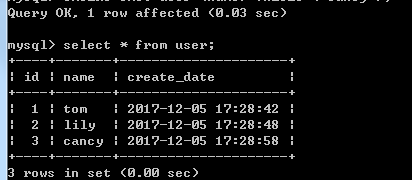
8）使用SHOW TABLES; 查看当前数据库下的表格

9）为user插入几条数据（可以使用可视化工具navicat插入）

INSERT INTO user (name) VALUES (‘tom’);

10）查看表格数据

SELECT \* FROM user;



**11、安装mysql依赖包**

npm install mysql --save

**12、连接mysql，创建增删改操作**

修改config/default.js文件，设置mysql连接参数：

module.exports = {

name:'znode',

port:5500,

mysql: {

connectionLimit:10,

host : 'localhost',

user : 'zyj',

password : '123',

database : 'znode'

}

}

新建lib文件夹，在下面添加sql.js用以连接数据库：

var config = require('config-lite')(\_\_dirname);

var mysql = require('mysql');

var pool = mysql.createPool(config.mysql);

function query(sql,callback){

pool.getConnection(function(err,conn){

if(err){

callback(err,null,null);

}else{

conn.query(sql,function(qerr,vals,fields){

//释放连接

conn.release();

//事件驱动回调

callback(qerr,vals,fields);

});

}

});

};

module.exports = query ;

创建models文件夹，创建user.js文件，用以操作user数据表

var query = require('../lib/sql');

module.exports = {

//获取所有

getAll:function(){

return new Promise(function(resolve,reject){

query('SELECT \* FROM user',function(err,val,fields){

if(err){

reject(err);

}else{

resolve({users:val,fields:fields})

}

})

})

}

}

**13、搭建前端框架，使用ajax操作请求数据**

使用正常的前端框架请求接口

如获取所有用户列表：

this.$http.get('http://192.168.4.131:5500/users/queryAll').then(function (res) {

if(res.data.code === '1'){

let data = res.data.result;

that.tData = data.splice(0);

}else{

let msg = res.data.message || '获取项目失败';

this.$message.error(msg);

}

}).catch(function (error) {

console.log(error);

});