北京智汇空间科技有限公司

产品设计文档

(Product / Feature Design Document)

室内导航定位服务器接口设计

|  |  |
| --- | --- |
| 文档编号： | DD-W001 |
| 文档版本： | V1.0 2018-04-18 |

室内导航定位服务器接口设计

# 前言

公司预测未来几年，室内定位将会进入大规模应用时期。基于BLE的室内定位技术是如今应用最广泛一项技术，发展潜力较大。由于所有的手机都带有BLE传感器，采用BLE定位技术进行室内导航定位有着天然的优势。在行业应用方面，使用BLE技术在无动力车定位方面也有较大优势。Node.js是一种高性能的异步事件服务器框架，内核采用C/C++编写，上层使用JavaScript封装和编程，支持大规模并发访问，综合分析比较适合BLE定位服务器的应用环境。室内导航定位系统如图所示。



# 总体设计

## 主要功能列表

|  |  |  |
| --- | --- | --- |
| 序号 | 功 能 | 添加日期 |
|  | 支持Windows，Linux，macOS | 20180718 |
|  | 采用HTTP Post协议 | 20180718 |
|  | 数据采用Json封装 | 20180718 |
|  | 支持1万用户在线 | 20180718 |
|  | 支持不同级别log文件记录 | 20180718 |
|  | 支持MySQL数据库 | 20180718 |
|  | 服务器参数可配置 | 20180718 |
|  | 服务器异常/错误捕获 | 20180718 |
|  | 支持多种权限访问 | 20180718 |
|  | 用户定位 | 20180718 |
|  | 用户跟踪 | 20180718 |
|  | 历史位置查询 | 20180718 |
|  | 用户注册 | 20180718 |
|  | 用户注销 | 20180718 |
|  | 用户登录/登出 | 20180718 |
|  | 设备管理 | 20180718 |
|  | 账户管理 | 20180718 |

## 设计原则

* 设计平台为：Visual Studio Code；
* 运行平台：Windows,Linux,macOS；
* 运行框架：Node.js
* 编程语言：JavaScript；
* 通信协议：HTTP；
* Web框架：express
* 打包软件：webpack
* 运行监控软件：pm2
* 数据接口协议： 见数据接口协议部分；
* 支持在线终端数量:10000；

## 软件流程图

### 总体流程图



### 用户管理



### 设备/账户管理



### 定位请求



### 跟踪请求



## 数据流程图



# 软件设计

## user.js

### 功能

用户类操作路由，包括用户登录、注册、注销接口。

### 接口

* api/user/login，登录
* api/user/regist，注册
* api/user/unregist，注销

### 方法

* user.login(object,callback)，登录接口调用方法
* user.regist(object,callback)，注册接口调用方法
* user.unregist(object,callback)，注销接口调用方法

### 应答

* 错误：错误信息
* 登录：用户配置信息
* 注册：默认用户配置信息
* 注销：成功信息

## manage.js

### 功能

设备和账号管理类路由，包括用户添加、更新、删除、查询接口。

### 接口

* api/manage/insert，添加
* api/manage/update，更新
* api/manage/delete，删除
* api/manage/select，查询

### 方法

* manage.insert(object,callback)，添加接口调用方法
* manage.update(object,callback)，更新接口调用方法
* manage.delete(object,callback)，删除接口调用方法
* manage.select(object,callback)，查询接口调用方法

### 应答

* 错误：错误信息
* 添加：成功信息
* 更新：成功信息
* 删除：成功信息
* 查询：设备或者账号信息

## track.js

### 功能

用户跟踪类路由，包括最新位置跟踪、历史位置查询。

### 接口

* api/track/latest，最新位置跟踪
* api/track/history，历史位置查询

### 方法

* track.latest(object,callback)，最新位置跟踪接口调用方法
* track.history(object,callback)，历史位置查询接口调用方法

### 应答

* 错误：错误信息
* 最新位置跟踪：查询用户最新位置信息
* 历史位置查询：查询用户历史位置信息

## locate.js

### 功能

位置请求类路由，包括位置存储、位置计算接口。

### 接口

* api/pos/save，位置存储
* api/pos/compute，位置计算

### 方法

* locate.location(object,callback)，位置接口调用方法

### 应答

* 错误：错误信息
* 位置存储：无应答
* 位置计算：位置信息

## log.js

### 功能

服务器log打印存储，监听log事件。

### 事件

* log：全局变量process监听

### 方法

* process.on(‘log’,callback)，监听log事件调用方法
* process.emit(‘log’,object)，发送log事件调用方法

## compute.js

### 功能

用户ble位置请求计算库接口。

### 方法

* ble(object,callback)，位置计算函数
* toHex(string)，字符串id转换数字id函数

### 返回

* err:object,错误信息
* pos:object,位置信息

## database.js

### 功能

服务器数据查询，添加，更新，删除等操作。

### 方法

* queryHasDestory(string,array,callback)，数据操作函数

### 返回

* err:object,错误信息
* results:object,结果信息

# 通信协议

所有数据接口采用HTTP协议的Post方式发送和接收数据。

# 数据接口协议

## 用户接口

### 登录

* 请求

{

"user": {

**"type": "user",**

**"id": "用户名",**

**"code": "密码"**

}

}

* 成功应答

{

"general": {

"type": "login",

"state":"true",

"message": "{数据库user\_config内容相同}"

}

}

* 失败应答

{

"general": {

"type": " login ",

"state":"flase",

"message": "失败原因"

}

}

### 注册

* 请求

{

"regist": {

**"type": "user",**

**"id": "用户名",**

**"code": "密码",**

"phone": "phone",

"email": "email",

"privilege": "user ",

"regist\_code": "注册码，管理员提供"

}

}

* 成功应答

{

"general": {

"type": "regist ",

"state":"true",

"message": "{数据库user\_config内容相同}"

}

}

* 失败应答

{

"general": {

"type": " regist ",

"state":"flase",

"message": "失败原因"

}

}

### 注销

* 请求

{

"user": {

**"type": "user ",**

**"id": "id",**

**"code": "password"**

}

}

* 成功应答

{

"general": {

"type": "unregist",

"state":"true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " unregist ",

"state": "flase",

"message": "失败原因"

}

}

## 设备管理接口

### 添加

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"insert": [

{

**"type": "beacon",**

**"id": "id",**

**"lati": 12.34512345,**

**"longi": 112.25478971,**

"accuracy": 12.445,

"alti": 4.256,

"alti\_accuracy": 0.256,

**"calibration": -61,**

"soc":99,

"location": "嘉华大厦",

"company": "smartspace",

"info": "test",

**"regist\_code": " regist\_code "**

}

]

}

* 成功应答

{

"general": {

"type": " insert",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " insert",

"state": "flase",

"message": "失败原因"

}

}

### 更新

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"update": [

{

**"type": "beacon",**

**"id": "id",**

"lati": 12.34512345,

"longi": 112.25478971,

"accuracy": 12.445,

"alti": 4.256,

"alti\_accuracy": 0.256,

"calibration": -61,

"soc":99,

"location": "嘉华大厦",

"company": "smartspace",

"info": "test",

"regist\_code": " regist\_code "

}

]

}

* 成功应答

{

"general": {

"type": " update ",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " update ",

"state": "flase",

"message": "失败原因"

}

}

### 删除

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"delete": {

**"type": "beacon ",**

**"id": ["\_\_all(全部)/id1,id2"]**

}

}

* 成功应答

{

"general": {

"type": " delete",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " delete ",

"state": "flase",

"message": "失败原因"

}

}

### 查询

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"select": {

**"type": "beacon ",**

**"id": ["\_\_all(全部)/id1,id2"]**

}

}

* 成功应答

{

"general": {

"type": " select ",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " select ",

"state": "flase",

"message": "失败原因"

}

}

## 账号管理接口

### 添加

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"insert": [

{

**"type": "user",**

**"id": "id",**

**"code": "password",**

"phone": "phone",

"email": "email",

"privilege": "user",

"regist\_code": "regist code",

"info":"info"

}

}

* 成功应答

{

"general": {

"type": " insert",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " insert",

"state": "flase",

"message": "失败原因"

}

}

### 更新

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"update": [

{

**"type": "user",**

**"id": "id",**

"code": "password",

"phone": "phone",

"email": "email",

"privilege": "user",

"regist\_code": "regist code",

"info":"info"

}

]

}

* 成功应答

{

"general": {

"type": " update ",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " update ",

"state": "flase",

"message": "失败原因"

}

}

### 删除

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"delete": {

**"type": " user ",**

**"id": ["\_\_all(全部)/id1,id2"]**

}

}

* 成功应答

{

"general": {

"type": " delete",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " delete ",

"state": "flase",

"message": "失败原因"

}

}

### 查询

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"select": {

**"type": " user ",**

**"id": ["\_\_all(全部)/id1,id2"]**

}

}

}

* 成功应答

{

"general": {

"type": " select ",

"state": "true",

"message": "success"

}

}

* 失败应答

{

"general": {

"type": " select ",

"state": "flase",

"message": "失败原因"

}

}

## 定位接口

### 计算/存储

* 请求

{

"user": {

**"type": "user",**

**"subType": "compute/save(无应答)",**

**"id": "id",**

**"code": "password"**

},

"beacon": [

{

**"id": "B9407F30F5F8466E",**

**"rssi": -60,**

"calibration": -45,

"soc": 30,

"scanTime": -333

}

]

}

* 成功应答

{

"pos": [

{

"id": "id",

"lati": 42.12345,

"longi": -117.23456,

"accuracy": 1.5,

"alti": 34.234,

"altitudeAccuracy": 2.5,

"heading": 30.8,

"velocity": 12.9

}

]

}

* 失败应答

{

"general": {

"type": " insert",

"state": "flase",

"message": "失败原因"

}

}

## 跟踪接口

### 最新位置/历史位置

* 请求

{

"user": {

**"type": "user",**

**"id": "id",**

**"code": "password"**

},

"track": {

**"type": "history/latest",**

**"deviceType": "user ",**

**"id": ["\_\_all/id"],**

"zone": [42.12345, 117.23456, 34.234, 42.12345, 117.23456, 34.234],

"timestamp": "2018-3-27 17:06:48@2018-3-27 17:06:48"

}

}

* 成功应答

{

"general": {

"type": " history/latest ",

"state": "true",

"message": "[{数据库user\_location内容相同}]"

}

}

* 失败应答

{

"general": {

"type": " history/latest ",

"state": "flase",

"message": "失败原因"

}

}

# 服务器配置

## 数据库

* 文件：config/database.json

{

"host": "localhost",

"user": "root",

"password": "smartspace",

"database": "smartspace",

"connectionLimit": 100,//连接数控制

"multipleStatements": true//多条语句同时执行

}

## 用户权限

* 文件：config/ privilege.json

{

"user": {

"isDataBase": true,//是否存储位置

"saveTime": 3,//存储时间

"queryTime": 1,//可查询时间

"compute": true//是否计算位置

},

"tracker": {

"isDataBase": true,

"saveTime": 6,

"queryTime": 3,

"compute": false

},

"admin": {

"isDataBase": true,

"saveTime": 12,

"queryTime": 12,

"compute": true

}

}

## 日志控制

* 文件：config/log.json

{

"isLog": true,//是否打印

"isAccess": true,//是否打印access

"isInfo": true,// 是否打印info

"isWarn": true,// 是否打印warn

"isError": true,// 是否打印error

"isDebug": false,// 是否打印debug

"level": 1,//1:全部打印，2:不打印message，3:仅打印codeNo

"code": true,// 是否打印code

"codeNo": true,// 是否打印codeNo

"type": true,// 是否打印type

"sysCall": true,// 是否打印sysCall

"message": true//是否打印message

}

* 文件：config/log4js.json

{

"appenders": {

"debug": {

"type": "file",//类型

"filename": "./logs/http/debug.log",//名称

"maxLogSize": 104857600,//大小

"numBackups": 3//备份数量

},

"access": {

"type": "file",

"filename": "./logs/http/access.log",

"maxLogSize": 104857600,

"numBackups": 20

},

"info": {

"type": "file",

"filename": "./logs/http/info.log",

"maxLogSize": 104857600,

"numBackups": 20

},

"error": {

"type": "file",

"filename": "./logs/http/errors.log",

"maxLogSize": 104857600,

"numBackups": 20

},

"default": {

"type": "file",

"filename": "./logs/http/default.log",

"maxLogSize": 104857600,

"numBackups": 3

}

},

"categories": {

"debug": {

"appenders": ["debug"],

"level": "DEBUG"

},

"info": {

"appenders": ["info"],

"level": "DEBUG"

},

"error": {

"appenders": ["error"],

"level": "DEBUG"

},

"access": {

"appenders": ["access"],

"level": "DEBUG"

},

"default": {

"appenders": ["default"],

"level": "DEBUG"

}

},

"pm2": true,

"disableClustering": true

}

## 运行监控

* 文件：config/ ecosystem.json（pm2软件配置文件）

{

"apps": [

{

"name": "ls-http-fork",//进程名

"watch": true,//文件变化自动重启

"ignore\_watch": ["node\_modules", "logs", "test","doc"],//忽略监控文件

"script": "./dist/www.js",//启动脚本

"error\_file": "./logs/pm2/err.log",//错误日志

"out\_file": "./logs/pm2/out.log",//普通日志

"log\_date\_format": "YYYY-MM-DD HH:mm:ss",//日志时间格式

"env": { "NODE\_ENV": "production", "PORT": 8002 }//运行环境和监听接口

},

{

"name": "ls-http-cluster",

"watch": true,

"ignore\_watch": ["node\_modules", "logs", "test","doc"],

"script": "./dist/www.js",

"instances": "max",//进程数服务器CPU个数相同

"exec\_mode": "cluster",//多进程模式

"error\_file": "./logs/pm2/err.log",

"out\_file": "./logs/pm2/out.log",

"log\_date\_format": "YYYY-MM-DD HH:mm:ss",

"env": { "NODE\_ENV": "production", "PORT": 8002 }

}

]

}

# 程序打包

* 打包软件

Webpack：用于将用户代码和第三方库代码进行打包发布

* 打包脚本

webpack.config.js：根据软件要求设置相关参数，运行npm run build进行打包，具体参数含义参考webpack官网文档。

const path = require('path')

module.exports = {

entry: { www: './src/http/bin/www.js' },//程序入口

output: {

filename: '[name].js',//发布程序名

path: path.resolve(\_\_dirname, 'dist')//发布位置

},

module: {

rules: [

{

test: /\.node$/,

use: 'node-loader'//C/C++接口打包支持

}

]

},

// mode: 'production',

mode: 'development',//打包模式

// devtool: 'inline-source-map',

target: 'node'//平台

}

* 打包修改

webpack.md：打包时需要进行相关修改，开发时可不修改。

1. Modified: log4js/lib/configuration.js

class Configuration/tryLoading

return require(modulePath)改为return require('./appenders/file')

2. Delete:express/lib/view.js

function View

删除var fn = require(mod).\_\_express

3. path:ls-api/src/lib/compute.js

改为绝对路径require('C:/node/ble.node')

4. path:ls-api/src/lib/compute.js

addon.ble()

改为绝对路径'C:/node/smartpos\_x64.dll'

5. Modified:ls-api/src/http/app.js

app.use(express.static(path.resolve(\\_\\_dirname, '../../../ls-vue/dist'))) 开发

app.use(express.static(path.resolve(process.cwd(), './dist'))) 生成

# 软件版本功能规划

表1 软件版本

|  |  |  |
| --- | --- | --- |
| **版本** | **功能** | **发布日期** |
| V0.1 | 正式发布前的最后内部测试版本，之后进入版本控制 | 20180718 |
| V1.0 |  |  |
| V2.0 |  |  |
| 。。。 |  |  |
| 注：随着软件版本的升级，本表同步更新。 | | |

1. 修订记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **文档编号** | **修订内容** | **修订部门** | **修订人** | **日期** |
| DD-V001-001\_V0.1 180718 | 文档初稿 | 技术部 | 毛刚 | 20180718 |
|  |  |  |  |  |
|  |  |  |  |  |