**Automatic Gardener User manual**

**Part 1 software**

1 Introduction

1.1 The purpose of the preparation

This document is the user manual written for the user in the early development phase of the intelligent flowerpot software system. In this document, we conduct detailed detailed analysis of the needs of different users, so that the reader understands all the functions of the software and the specific permissions of the user. The

1.2 Project Background

The project production unit: International Education College of Zhengzhou University of Light Industry & IGP project team of Edinburgh Napier University.

1.3 Definition

RN: Facebook's open source cross-platform mobile application development framework in April 2015, currently supports both iOS and Android platforms.

SQL: Structured Query Language structured language.

Arduino: Arduino is an open source electronic prototype platform that is easy, flexible, and easy to use. Includes hardware (various models of Arduino boards) and software (Arduino IDE).

JS (JavaScript): JavaScript A direct translation scripting language, a dynamic type, weak type, prototype-based language, built-in support type.  
1.4 References

1. https://www.arduino.cn/thread-7540-1-1.html - Esp8266wifi module
2. [https://github.com/devstepbcn/react-native-android-wifi/blob/m aster/README.md](https://github.com/devstepbcn/react-native-android-wifi/blob/master/README.md) - github wifi module
3. <https://github.com/ccmxy/react-native-wifi-module/blob/master/README.md-github> shared

2. Software Overview

2.1 Objectives: To achieve automatic watering, measurement of soil moisture, light intensity, plant status, and other information, and plant naming and display plant database information and other functions.

2.2 Functions

   a. Automatic watering

b. Plant Name

C. Measuring soil temperature, humidity, nutrients, etc.

D. Compare with other plants

2.3 Performance

a. Automatically measure when watering (accurate to seconds), measure plant nutrition and moisture.

b. Response time: about 5 seconds. Working time: 10 seconds or so. Sleep time: The system automatically enters sleep mode when it is not working for five minutes.

c. Flexibility Solar panels: In addition to the normal power supply model, we also produced a solar panel as an additional power supply function. Sleep Mode: Since the motherboard is very power consuming, we have set the sleep mode so that the system automatically turns off when it does not work for five minutes, minimizing power consumption.

3. Operating environment

3.1 Hardware

a a smart phone

b Plants

C Smart Flower Pot

3.2 Support Software

The application software development language of the smart flower pot is JavaScript, the framework technology used is React Native, the development tool used is VS code, the software developed based on the third party library is Expo, and one of the biggest features of React Native is cross platform. Therefore, the operating system for developing the App uses MacOS/Linux/Windows. For faster development, the development platform uses a Mac system and the target platform is Android. The React Native software developed can be run on the Expo software in the Android (6.0+)/IOS (9.0+) dual-platform.

Since the software is developed using JavaScript language, JS as a static scripting language does not require any compilation during the development process. After the relevant code is modified, the result can be displayed on the Android device or the emulator as long as it is saved and refreshed.

The database software used by this app is: SQLite3, version number is 3.23.1, the biggest characteristic of this database is extremely lightweight, and the operation is simple and convenient.

**Other necessary support software: (Development platform: MacOS target platform: Android)**

**1Homebrew -- Mac system's package manager for installing NodeJS and some other necessary tools.**

**2Node -- React Native currently requires NodeJS 5.0 or later. After installing node, it is recommended to set up npm mirroring to speed up the following process (or use scientific online tools)**

**3Yarn, React Native's command-line tool (react-native-cli) Yarn is Facebook's alternative to npm and can speed up the download of node modules. React Native's command line tools are used to perform tasks such as creating, initializing, updating projects, running packager, and so on.**

**4Android Studio——(React Native currently requires Android Studio 2.0 or higher. Android Studio requires Java Development Kit [JDK] 1.8 (later not supported later).)**

**5Watchman -- A tool provided by Facebook to monitor file system changes. Install this tool to improve development performance (packager can quickly capture file changes for real-time refresh)**

**6Flow -- is a static JS type checking tool**

**7Git————Git version control**

**8 Genymotion - Genymotion is a better choice than the original simulator that comes with Android Studio.**

**9Gradle Daemon --- Open Gradle Daemon can greatly increase the incremental compilation speed of java code**

**4. Instructions for use**

**4.1 Installation and Initialization**

**The program is stored in binary form;**

**Operation command: 1npm install 2exp start in Command Line;**

**Feedback: The Bundle JavaScript% progress bar is displayed on the Expo software in the mobile device or Genymotion. The progress bar shows the progress of the software installation. If the screen is red or blue, and there is an error message on the screen, a problem with the code indicates an error.**

**Install the required software tools: Need the latest version of Expo**

**4.2 Input**

**1 string: is a variable that stores characters.**

**Eg:var flowername="Rose";**

**2 Number types: Numbers can be with or without decimal point.**

**Eg: var x1=34.00; // use the decimal point to write**

**Var x2=34; // Don't use decimal point to write**

**3 Boolean (logical): There can be only two values: true or false.**

**Eg:var x=true**

**Var y=false**

**4 array**

**Eg:var flowers=new Array("Rose","peony","lily");**

**5 objects: separated by braces. Inside the brackets, the properties of the object are defined in the form of name : value pairs.**

**Eg: var flower={name:"Rose", temperature:"30°", id:123};**

**6Undefined and Null: Undefined This value indicates that the variable does not contain a value. You can clear the variable by setting its value to null.**

**Eg: flowers=null;**

**4.2.1 Data Background**

**Source: Baidu/Google**

**Storage Media: SQLite3 Database**

**Occurrence frequency: higher**

**Restrictions and quality management:? ? ?**

**4.2.2 Data Format**

**a. Length: no limit**

**b. Format benchmark: utf-8**

**c. Separator: Split by ""**

4.3 output

Dynamic output data: A plant light intensity B plant soil nutrient C water level height D plant dry and wet state (watering or not.)

4.3.l data background

A React-native framework B data interface C database analysis

4.3.2 Data Format

Software through wireless routing, the hardware log information is json format. Similar to {

"data":[

{"time":"0000","power":"100"}

{"time":"0005","power":"100"}

{"time":"0010","power":"100"}

]

}

4.4 Errors and Recovery

Error: The automatic watering button will not work when there is a problem with the software. The warning screen appears.

Recovery: Check if the hardware and water are sufficient and restart the software. The

4.5 help query

When there is a problem, the user can choose to upload the error message. The background will be recorded so that program personnel can perform maintenance. 

4.4 Errors and Recovery

Error: The automatic watering button will not work when there is a problem with the software. The warning screen appears.

Recovery: Check if the hardware and water are sufficient and restart the software. The

4.5 help query

When there is a problem, the user can choose to upload the error message. The background will be recorded so that program personnel can perform maintenance.

5. Operation instructions

5.1 Operation Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Interface Display | Display Contents | Operational Process | Purpose | Estimated Operating Time | Format and Format of Operation Commands | Operational Requirements |
| Automatic watering display | Plants need watering | Hardware detection of plant status - Transfer to back-end database - Calculate plant status and judge plants need watering - Human-computer interaction | Keep the plant's water balance and prevent plant death. | Watering time: 5 seconds  Background calculation time: five seconds to fifteen seconds | The data saved in the log file is returned via hardware using Json | 1. Need to have a valid 2.object (plant).   1. Need users to turn onapp。 2. The user is required to open the network connection. |
| Automatic watering display off | Plants do not require watering | Hardware detection of plant status - Transfer to back-end database - Calculate plant status and judge plant does not require watering - Human-computer interaction | When watering is not required, information show on the plant (such as soil moisture, light intensity) is displayed so that the user can better understand the state of the plant. | None | The data saved in the log file is returned via hardware using Json | 4. Need to have a valid object (plant).  5. The user is required to open the app.  Require user to open network connection。 |

6****. Operation command list****

**1, open the command line window**

**#db2cmd**

**2, open the control center**

**# db2cmd db2cc**

**3, open the command editor**

**Db2cmd db2ce**

**===== Operational database command =====**

**4, start the database instance**

**#db2start**

**5, stop the database instance**

**#db2stop**

**If you cannot stop the database due to the active connection, execute db2 force application all before running db2stop /db2stop force**

**6, create a database**

**#db2 create db [dbname]**

**7, connect to the database**

**#db2 connect to [dbname] user [username] using [password]**

**8, disconnect the database connection**

**#db2 connect reset**

**9, list all databases**

**#db2 list db directory**

**10. List all active databases**

**#db2 list active databases**

**11. List all database configurations**

**#db2 get db cfg**

**12, delete the database**

**#db2 drop database [dbname]**

**(Be careful when doing this)**

**If you cannot delete, disconnect all database connections or restart db2**

**========= Operation Data Table Command ==========**

**13, list all user tables**

**#db2 list tables**

**14, list all system tables**

**#db2 list tables for system**

**15, list all tables**

**#db2 list tables for all**

**16, list the system table**

**#db2 list tables for system**

**17, list the user table**

**#db2 list tables for user**

**18, list a specific user table**

**#db2 list tables for schema [user]**

**19. Create a new table (t1) with the same structure as a table (t2) in the database**

**#db2 create table t1 like t2**

**20, the data of one table t1 is imported into another table t2**

**#db2 "insert into t1 select \* from t2"**

**21, the query table**

**#db2 "select \* from table name where ..."**

**22, display table structure**

**#db2 describe table tablename**

**23, modify the column**

**#db2 alter table [tablename] alter column [columname] set data type varchar(24)**

**====== Script file operation command =======**

**24, execute the script file**

**#db2 -tvf scripts.sql**

**25, help command**

**\* See command help**

**#db2 ? db2start**

**\* View error code information**

**#db2 ? 22001**

**\* memo: Please use "db2 ? <command>" for detailed commands. The**

**=========================**

**26, backup the database**

**#db2 backup db <db name>**

**Note: need to disconnect the database before executing the above command**

**27, online backup database**

**#db2 -v "BACKUP DATABASE <database name> ONLINE TO <path> WITH 2 BUFFERS BUFFER 1024 INCLUDE LOGS WITHOUT PROMPTING"**

**28, restore the database**

**#db2 restore db <source db name>**

**29, online recovery database**

**#db2 "RESTORE DB <database name> TO <db path> LOGTARGET <logpath> WITHOUT PROMPTING"**

**#db2 "ROLLFORWARD DB <database name> TO END OF LOGS AND STOP" ...**

**30, export data files**

**#db2move <db name> export**

**[-sn <schema name, typically db2admin>]**

**[-tn <table name, separated by multiple commas>]**

**31, import data files**

**#db2move <db name> import**

**32, get db2 database management configuration environment information**

**#db2 get dbm cfg**

**33. Get db2 database database management configuration environment information**

**#db2 get db cfg for <db name>**

**Or: Execute db2 get db cfg after connecting to a database**

**34, change the size of db2 log space**

**Note: The following command is used to prevent the db2 database from using excessive hard disk space. It is only used for db2 on the developer's own machine. If it is a server, the parameters need to be modified.**

**#db2 UPDATE DB CFG FOR <db name> USING logretain OFF logprimary 3 logsecond 2 logfilsiz 25600;**

**If the page size is 4KB, the above command creates three 100M log files, occupying 300MB of hard disk space. 25600\*4KB=102400KB.**

**35, create a temporary table space**

**#DB2 CREATE USER TEMPORARY TABLESPACE STMASPACE PAGESIZE 32 K MANAGED BY DATABASE USING (FILE 'D:\DB2\_TAB\STMASPACE.F1' 10000)**

**EXTENTSIZE 256**

**36, get the database manager snapshot data**

**#db2 –v get snapshot for dbm**

**37, display the process number**

**#db2 list applications show detail**

**The**

**8. List of program files (or command files) and data files**

**[List file names, identifiers, and descriptions alphabetically by file name or by function and module sort order. 】**

Front-end management:

Step 1: Log in to the software. If you have an account, log in to your account. If you do not have an account, click the Register button.

Step 2: After entering the main interface, click the Add Plant button, add your plant and name the plant.

The third step: After adding the plant, wait for the system to measure the plant information. If the information is wrong, you can choose to add the plant again.

The fourth step is to keep the mobile phone network open and automatically water when the system measures that the plants need to be watered.

**Part 2 hardware**

1 Introduction

1.1 The purpose of the preparation

This document is the user manual written by the user in the early stage of development of the smart pot hardware. In this document, we conduct detailed detailed analysis of the needs of different users, allowing the user to have a general understanding of the hardware system, and detailed description of the hardware Composition and problem solutions. The

1.2 Project Background

The project production unit: International Education College of Zhengzhou University of Light Industry & IGP project team of Edinburgh Napier University.

1.3 Definition

Arduino: Arduino is an open source electronic prototype platform that is easy, flexible, and easy to use. Includes hardware (various models of Arduino boards) and software (Arduino IDE).

ESP32/ESP07: The ESP32 can be used as a stand-alone system to run applications or slave devices of the host MCU, providing WIFI and Bluetooth functions via SPI/SDIO or I2C/UART interfaces. ESP07: Developed by Infosys Technologies, the module's core processor, the ESP8266, integrates an industry-leading MCU in a smaller package with 16-bit thin mode.

Water Level/Soil Humidity/Light Intensity Sensors: These three sensors are detection equipments that can sense the measured information and can change the sensed information into electric signals or other desired forms of information output according to certain rules. Meet project hardware requirements.

Solar panels: Solar panels are devices that directly or indirectly convert electricity into solar energy by absorbing sunlight and converting the solar radiation through photoelectric or photochemical effects.

Water pump: A machine that transfers liquid or pressurizes a liquid and transfers the mechanical energy of the prime mover or other external energy to the liquid, causing the liquid energy to increase.

1.4 References

4.https://www.arduino.cn/thread-7540-1-1.html - Esp8266wifi module

5.https://github.com/devstepbcn/react-native-android-wifi/blob/m aster/README.md - github wifi module

6.https://github.com/ccmxy/react-native-wifi-module/blob/master/README.md-github Share

The

2. Hardware Overview

2.1 Objectives Successfully connected to the software wirelessly and controlled the water pump to automatically water the plants. Each sensor can successfully measure plant information and display success on the mobile phone software.

2.2 Functions

  a. Automatic watering

b. Solar panels

C. Each sensor measures plant information

D.3D flower pot real model

2.3 Performance

a. Dual-power mode: The battery and the solar panel will use the solar energy stored in the solar panel when the battery is not powered.

b. Fully automated operation: The user only needs to place the plant in the flowerpot. The flowerpot will automatically measure the plant information and the operation is very friendly.

C. Sleep Mode: The sleep mode is set in the hardware background, that is, when there is no information output for five minutes, the module will automatically shut down and enter sleep mode to effectively reduce the power consumption.

3. Operating environment

3.1 Hardware

1.3D flower pot model

2. Plant

3. Water pump

4. Instructions for use

4.1 Installation and Initialization

/hello

GET

No parameters are required

Return :

200 OK

Application/ Json

{

"ChipID": "XXXXXXXXX",

"MAC": "DDDDDDDDDD"

}This route is used to confirm that the device on the IP is an intelligent flowerpot.

It is used to confirm that the device on this IP is a smart flowerpot.

/reset

POST

Arg1: ppass value: (Set of flowerpot ciphers)

OK return: 200 OK

Fail return : 400 Bad Request

This route is used to reset the WIFI account and password. PPASs must be input before reset, and PPASs will be reset after input.

This route is used to reset the WIFI account and password. Before the reset, you must enter ppass. After the reset, ppass will be reset.

The

4.2.2 Data Format

a Json is constructed in two structures:

 A collection of name/value pairs. In different languages, it is understood to be an object, a record, a struct, a dictionary, a hash table, a keyed list, or an associative array. Array).

 An ordered list of values. In most languages, it is understood as an array.

These are common data structures. In fact, most modern computer languages ​​support them in some form. This makes it possible to exchange a data format between programming languages ​​that are also based on these structures.

b JSON has the following form:

The object is an unordered collection of 'name/value pairs'. An object starts with "{" (left parenthesis) and "}" (right parenthesis) ends. Each "name" is followed by a ":" (colon); ""Name/value pairs" are separated by "," (comma).



An array is an ordered collection of values. An array starts with "[" (left bracket) and "]" (right bracket) ends. Separate values with "," (comma).



The value can be a double quoted string, number, true, false, null, object, or array. These structures can be nested.



A string is a set of any number of Unicode characters surrounded by double quotes, escaped with a backslash. A character is a single character string.

Strings are very similar to C or Java strings.

The number is also very similar to the value of C or Java. Remove unused octal and hexadecimal formats. Remove some coding details.



4.2.3 Example of input

4.3 output

Dynamic output data: Data measured by each sensor

Code:

/log

GET

Arg1: date value1: (yyyyMMdd类型字符串)

OK return: 200 OK (application/json)

e.g.

{

"data": [{

"time": "0100",

"power": "3.7",

"soil": "1024",

"temp": "25.0",

"humidity": "35.2",

"waterlevel": "5",

"light": "1024"

},

{

"time": "0105",

"power": "3.7",

"soil": "1024",

"temp": "25.0",

"humidity": "35.2",

"waterlevel": "5",

"light": "1024"

},

{

"time": "0110",

"power": "3.7",

"soil": "1024",

"temp": "25.0",

"humidity": "35.2",

"waterlevel": "5",

"light": "1024"

},

{

"time": "0115",

"power": "3.7",

"soil": "1024",

"temp": "25.0",

"humidity": "35.2",

"waterlevel": "5",

"light": "1024"

}

]

}

Fail return: 404 Not Found (空页面)

4.3.2 Data Format

* 8th:
  + [json>](https://8th-dev.com/json.html).
* ABAP:
  + [EPO Connector](http://www.epoconsulting.com/index.php/en/solution/sap-integration).
* ActionScript:
  + [ActionScript3](http://github.com/mikechambers/as3corelib).
* Ada:
  + [GNATCOLL.JSON](http://docs.adacore.com/gnatcoll-docs/json.html).
* AdvPL:
  + [JSON-ADVPL](https://github.com/imsys/JSON-ADVPL).
* ASP:
  + [JSON for ASP](https://github.com/tugrul/aspjson).
  + [JSON ASP utility class](http://www.webdevbros.net/2007/04/26/generate-json-from-asp-datatypes/).
* AWK:
  + [JSON.awk](https://github.com/step-/JSON.awk).
  + [rhawk](https://github.com/Andy753421/rhawk/blob/master/json.awk).
* Bash:
  + [Jshon](http://kmkeen.com/jshon/).
  + [JSON.sh](https://github.com/dominictarr/JSON.sh).
* BlitzMax:
  + [bmx-rjson](https://github.com/Trylobot/bmx-rjson).
* C:
  + [JSON\_checker](https://github.com/douglascrockford/JSON-c).
  + [YAJL](http://lloyd.github.com/yajl/).
  + [LibU](http://github.com/koanlogic/libu/blob/master/srcs/toolbox/json.c).
  + [json-c](https://github.com/jehiah/json-c).
  + [json-parser](https://github.com/udp/json-parser).
  + [jsonsl](https://github.com/mnunberg/jsonsl).
  + [WJElement](https://github.com/netmail-open/wjelement).
  + [M's JSON parser](http://sourceforge.net/projects/mjson/).
  + [cJSON](http://github.com/DaveGamble/cJSON).
  + [Jansson](http://www.digip.org/jansson/).
  + [jsmn](https://bitbucket.org/zserge/jsmn/wiki/Home).
  + [parson](http://kgabis.github.com/parson/).
  + [ujson4c](https://github.com/esnme/ujson4c/).
  + [nxjson](https://bitbucket.org/yarosla/nxjson/src).
  + [frozen](https://github.com/cesanta/frozen).
  + [microjson](http://www.catb.org/esr/microjson/).
* C++:
  + [JSONKit](https://github.com/johnezang/JSONKit).
  + [jsonme--](https://github.com/pibara/jsonme--).
  + [ThorsSerializer](https://github.com/Loki-Astari/ThorsSerializer).
  + [JsonBox](https://github.com/anhero/JsonBox).
  + [jvar](https://github.com/YasserAsmi/jvar).
  + [rapidjson](https://github.com/miloyip/rapidjson).
  + [JSON for Modern C++](https://github.com/nlohmann/json).
  + [ArduinoJson](https://github.com/bblanchon/ArduinoJson).
  + [minijson](https://giacomodrago.github.io/minijson/).
  + [jsoncons](https://github.com/danielaparker/jsoncons).
  + [QJson](http://qjson.sourceforge.net/).
  + [jsoncpp](http://sourceforge.net/projects/jsoncpp/).
  + [JOST](http://ddsbench.svn.sourceforge.net/viewvc/ddsbench/trunk/jost/).
  + [CAJUN](http://cajun-jsonapi.sourceforge.net/).
  + [libjson](http://sourceforge.net/projects/libjson/).
  + [nosjob](http://fossil.wanderinghorse.net/repos/nosjob/index.cgi/index).
  + [JSON++](https://bitbucket.org/tunnuz/json).
  + [JSON library for IoT](https://realtimelogic.com/products/json/).
  + [qmjson](https://github.com/QtMark/qmjson).
  + [JSON Support in Qt](http://doc.qt.io/qt-5/json.html).
  + [JsonWax for Qt](https://doublejim.github.io/).
* C#:
  + [fastJSON](http://www.codeproject.com/KB/IP/fastJSON.aspx).
  + [JSON\_checker](http://www.raboof.com/projects/jsonchecker/).
  + [Jayrock](https://bitbucket.org/raboof/jayrock/wiki/Home).
  + [Json.NET - LINQ to JSON](http://james.newtonking.com/projects/json-net.aspx).
  + [LitJSON](http://lbv.github.io/litjson/).
  + [JSON for .NET](http://sourceforge.net/projects/csjson).
  + [JSON@CodeTitans](http://codetitans.codeplex.com/).
  + [JSONSharp](http://code.google.com/p/jsonsharp/).
  + [fluent-json](http://code.google.com/p/fluent-json/).
  + [Manatee Json](https://github.com/gregsdennis/Manatee.Json).
  + [FastJsonParser](https://github.com/ysharplanguage/FastJsonParser).
  + [LightJson](https://github.com/MarcosLopezC/LightJson).
  + [liersch.json](https://github.com/steffen-liersch/liersch.json).
* Ciao:
  + [Ciao JSON encoder and decoder](http://ciaohome.org/docs/ciao/json.html).
* Clojure:
  + [data.json](http://clojure.github.com/data.json/).
* Cobol:
  + [XML Thunder](http://www.xmlthunder.com/).
  + [Redvers COBOL JSON Interface](http://www.redversconsulting.com/cobol_json_interface.php).
* ColdFusion:
  + [SerializeJSON](http://help.adobe.com/en_US/ColdFusion/10.0/CFMLRef/WSc3ff6d0ea77859461172e0811cbec22c24-79fa.html).
  + [toJSON](http://tojson.riaforge.org/).
* D:
  + [Libdjson](https://256.makerslocal.org/wiki/index.php/Libdjson).
* Dart:
  + [json library](http://api.dartlang.org/json.html).
* Delphi:
  + [Delphi Web Utils](https://sourceforge.net/projects/is-webstart/).
  + [JSON Delphi Library](http://sourceforge.net/projects/lkjson).
* E:
  + [JSON in TermL](http://erights.org/data/terml/embeddings.html).
* Fantom:
  + [Json](http://fantom.org/doc/docLib/Json.html).
* FileMaker:
  + [JSON](http://www.modularfilemaker.org/module/json/).
* Fortran:
  + [json-fortran](https://github.com/jacobwilliams/json-fortran).
  + [YAJL-Fort](http://sourceforge.net/projects/yajl-fort/).
* Go:
  + [package json](http://golang.org/pkg/encoding/json/).
* Groovy:
  + [groovy-io](https://github.com/jdereg/groovy-io).
* Haskell:
  + [RJson package](http://hackage.haskell.org/cgi-bin/hackage-scripts/package/RJson).
  + [json package](http://hackage.haskell.org/cgi-bin/hackage-scripts/package/json).
* Java:
  + [JSON-java](https://github.com/stleary/JSON-java).
  + [JSONUtil](https://github.com/billdavidson/JSONUtil).
  + [jsonp](https://java.net/projects/jsonp/).
  + [Json-lib](http://json-lib.sourceforge.net/).
  + [Stringtree](http://www.stringtree.org/stringtree-json.html).
  + [SOJO](http://sojo.sourceforge.net/).
  + [json-taglib](http://json-taglib.sourceforge.net/).
  + [Flexjson](http://flexjson.sourceforge.net/).
  + [JON tools](http://sourceforge.net/projects/jontools/).
  + [Argo](http://argo.sourceforge.net/).
  + [jsonij](https://bitbucket.org/jmarsden/jsonij).
  + [fastjson](http://sourceforge.net/projects/fastjson/).
  + [mjson](http://bolerio.github.io/mjson/).
  + [jjson](https://github.com/grobmeier/jjson).
  + [json-simple](https://github.com/fangyidong/json-simple).
  + [json-io](https://github.com/jdereg/json-io).
  + [JsonMarshaller](http://code.google.com/p/jsonmarshaller/).
  + [google-gson](http://code.google.com/p/google-gson/).
  + [Json-smart](http://code.google.com/p/json-smart/).
  + [FOSS Nova JSON](https://github.com/fossnova/json).
  + [Corn CONVERTER](https://sites.google.com/site/javacornproject/corn-converter).
  + [Apache johnzon](http://johnzon.incubator.apache.org/).
  + [Genson](http://owlike.github.io/genson/).
  + [JSONUtil](http://kopitubruk.org/JSONUtil/).
  + [cookjson](https://github.com/coconut2015/cookjson).
* JavaScript:
  + [JSON](http://www.ecma-international.org/publications/standards/Ecma-262.htm).
  + [json2.js](https://github.com/douglascrockford/JSON-js).
  + [clarinet](https://github.com/dscape/clarinet).
  + [Oboe.js](http://oboejs.com/).
* LabVIEW:
  + [flatten](http://zone.ni.com/reference/en-XX/help/371361K-01/glang/flat_unflat_string/).
* Lisp:
  + [Common Lisp JSON](http://common-lisp.net/project/cl-json/).
  + [Emacs Lisp](http://edward.oconnor.cx/elisp/json.el).
* LiveCode:
  + [mergJSON](https://github.com/montegoulding/mergJSON).
* LotusScript:
  + [JSON LS](http://www.openntf.org/Projects/pmt.nsf/ProjectHome?ReadForm&Query=JSON LS).
* LPC:
  + [Grimoire: LPC JSON](http://lostsouls.org/grimoire_json).
* Lua:
  + [JSON Modules](http://lua-users.org/wiki/JsonModules).
* M:
  + [DataBallet](https://github.com/lparenteau/DataBallet/blob/master/r/json.m).
* Matlab:
  + [JSONlab](http://iso2mesh.sourceforge.net/cgi-bin/index.cgi?jsonlab).
  + [20565](http://www.mathworks.com/matlabcentral/fileexchange/20565).
  + [23393](http://www.mathworks.com/matlabcentral/fileexchange/23393).
* Net.Data:
  + [netdata-json](https://github.com/blackchair/netdata-json).
* Nim:
  + [Module json](https://nim-lang.org/docs/json.html).
* Objective C:
  + [NSJSONSerialization](https://developer.apple.com/library/ios/documentation/foundation/reference/nsjsonserialization_class/Reference/Reference.html).
  + [json-framework](https://github.com/stig/json-framework/).
  + [JSONKit](https://github.com/johnezang/JSONKit).
  + [yajl-objc](https://github.com/gabriel/yajl-objc).
  + [TouchJSON](https://github.com/TouchCode/TouchJSON).
* OCaml:
  + [Yojson](http://martin.jambon.free.fr/yojson.html).
  + [jsonm](http://erratique.ch/software/jsonm).
* PascalScript:
  + [JsonParser](https://github.com/koldev/JsonParser).
* Perl:
  + [CPAN](http://search.cpan.org/search?query=JSON).
  + [perl-JSON-SL](https://github.com/mnunberg/perl-JSON-SL).
* Photoshop:
  + [JSON Photoshop Scripting](http://www.tonton-pixel.com/blog/json-photoshop-scripting/).
* PHP:
  + [PHP 5.2](http://www.php.net/releases/5_2_0.php).
* PicoLisp:
  + [picolisp-json](https://github.com/aw/picolisp-json).
* Pike:
  + [Public.Parser.JSON](http://modules.gotpike.org/module_info.html?module_id=33).
  + [Public.Parser.JSON2](http://modules.gotpike.org/module_info.html?select=0.2&module_id=43).
* PL/SQL:
  + [pljson](https://github.com/pljson/pljson).
* PowerShell:
  + [PowerShell](http://www.codeplex.com/PowerShellJSON).
* PureBasic:
  + [JSON](http://www.purebasic.com/documentation/json/index.html).
* Puredata:
  + [PuRestJson](https://github.com/residuum/PuRestJson).
* Python:
  + [The Python Standard Library](http://docs.python.org/library/json.html).
  + [simplejson](http://pypi.python.org/pypi/simplejson/).
  + [pyson](http://jyson.xhaus.com/).
  + [Yajl-Py](http://pykler.github.com/yajl-py/).
  + [ultrajson](https://github.com/esnme/ultrajson).
  + [metamagic.json](http://pypi.python.org/pypi/metamagic.json/).
* R:
  + [rjson](http://cran.r-project.org/web/packages/rjson/index.html).
  + [jsonlite](http://cran.r-project.org/web/packages/jsonlite/index.html).
* Racket:
  + [json-parsing](http://www.neilvandyke.org/racket-json-parsing/).
* Rebol:
  + [json.r](http://www.rebol.org/view-script.r?script=json.r).
* RPG:
  + [JSON Utilities](http://rpgnextgen.com/index.php?content=json).
* Rust:
  + [Serde JSON](https://github.com/serde-rs/json).
  + [json-rust](https://github.com/maciejhirsz/json-rust).
* Ruby:
  + [json](http://json.rubyforge.org/).
  + [yajl-ruby](http://github.com/brianmario/yajl-ruby).
  + [json-stream](https://github.com/dgraham/json-stream).
  + [yajl-ffi](https://github.com/dgraham/yajl-ffi/tree/master/lib/yajl/ffi).
* Scheme:
  + [MZScheme](http://www.lshift.net/blog/2005/08/22/json-for-mzscheme-and-a-portable-packrat-parsing-combinator-library).
  + [PLT Scheme](http://planet.plt-scheme.org/display.ss?package=json.plt&owner=dherman).
* Squeak:
  + [Squeak](http://map1.squeakfoundation.org/sm/package/d38bdc2d-e52a-4167-ae73-2cf438c65c2f).
* Symbian:
  + [s60-json-library](http://code.google.com/p/s60-json-library/).
* Tcl:
  + [JSON](http://wiki.tcl.tk/13419).
* Visual Basic:
  + [VB-JSON](http://www.ediy.co.nz/vbjson-json-parser-library-in-vb6-xidc55680.html).
  + [PW.JSON](http://www.pozzware.com/pozzware/Corsi/Programmazione/VB.NET/JSON Library.aspx).
  + [.NET-JSON-Transformer](https://github.com/dday9/.NET-JSON-Transformer).
* Visual FoxPro:
  + [fwJSON](http://www.foxweb.com/document/index.htm?page=/document/fwJSONObject.htm).
  + [JSON](http://www.sweetpotatosoftware.com/SPSBlog/PermaLink,guid,5f96efd7-f350-4bc8-86c5-caffd9e833eb.aspx).
  + [vfpjson](https://github.com/sait/vfpjson).

4.4 Errors and Recovery

Error:

1. Sensor data read error.

2. The wrong priority

3. Can not automatically water (water pump does not work)

4. Inaccurate data measurement

Recovery: 1. Restart the ESP32/07 or relocate the sensor

2. Reconnect the wrong original.

3. Check the battery for power, check if the pump is working properly, restart the machine, and reconnect the device.

4. Check that the sensor is working properly. The

4.5 help query

When there is a problem with the hardware, the user can view the user's manual. If the problem cannot be solved, he can ask the professional to perform maintenance.  
  
****5. Operation instructions****

****5.1 Operation Table****

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Interface Display | Display Contents | Operational Process | purpose | Estimated Operating Time | Format and Format of Operation Commands | Operational Requirements |
| Automatic watering | ESP32/07Flashing light | Back-end calculations of plant data--determining that the plant needs to be watered--the main board gives the signal - the water pump starts to work - completes the watering process | Keep the plant's water balance and prevent plant death. | Watering time: 5 seconds  Background calculation time: five seconds to fifteen seconds | The data saved by the log file is returned in json when returned by hardware | 1. Need to have a valid object (plant).  2.The user is required to read the instruction manual.  3.You need to make sure that all parts work properly. |
|  | ESP32/07The light is red | The back-end calculates the plant data—determines that the plant does not require watering—the main board gives the signal—each sensor passes the data to the mobile app—the app displays plant information. | When watering is not required, information on the plant (such as soil moisture, light intensity) is displayed so that the user can better understand the state of the plant. | None | The data saved by the log file is returned in json when returned by hardware | 1.Need to have a valid object (plant).  2.The user is required to read the instruction manual.  Need to ensure that all parts work properly |

****6.Operation command list**** /AP

POST

Arg1: ssid value1: (0-32 byte string)

Arg2:pass value2: (8-64 byte string. If the length is less than 8, the output is empty.)

Arg3:ppass value3: (The 6 digit, must exist)

OK return: 200 OK (Empty page)

Fail return: 400 Bad Request (Empty page)

This routing occurs only in the AP mode. It is used to set up access points connected to the Internet and password to log in to flower pots.

settings

POST

Arg1: isAuto value1: (1、0)

Arg2: NTP value2: (字符串类型)

Arg3: ppass value3: (6位数字，必须有)

Arg4: timeZone value4:String;

OK return: 200 OK (空页面)

Fail return: 400 Bad Request (空页面)

This route is used to receive settings from the mobile phone

Can be set automatically for water

And the NTP server for the time

r

/plants

POST

Arg1: tempH value1 : (整数)

Arg2: tempL value2 : (整数)

Arg3: lightH value3: (整数)

Arg4: lightL value4: (整数)

Arg5: soilH value5: (整数)

Arg6: soilL value6: (整数)

Arg7: waterT value7: (整数)

Arg8: humidity value8: (整数)

Arg9: ppass value9: (6位数字，必须有)

OK return : 200 OK (空页面)

Fail return : 400 Bad Request (空页面)

This route is used to set up the plant related environment, which can be used as a reference for automatic watering.

**#Arduino Syntax**

**The**

**Setup() initialization function**

**The**

**Loop() loop body function**

**The**

**Control statements are similar to C**

**//if if...else for switch case while do... while break continue return goto**

**The**

**Extended syntax similar to C**

**// (semicolon) {} (brace) // (single-line comment) /\* \*/ (multi-line comment) #define #include**

**The**

**The arithmetic operator is similar to C**

**//=(assignment operator) + (plus) - (minus) \* (multiply) / (divide) % (module)**

**The**

**Comparison operators are similar to C**

**//==(equals) !=(not equals) <(less than) >(greater than) <=(less than or equals) >=(greater than or equals)**

**The**

**Boolean operators are similar to C**

**//&&(and) || (or) ! (not)**

**The**

**Pointer operators are similar to C**

**//\* dereference operator & reference operator**

**The**

**Bit operators are similar to C**

**& (bitwise and) | (bitwise or) ^ (bitwise xor) ~ (bitwise not) << (bitshift left) >> (bitshift right)**

**The**

**Compound operator is similar to C**

**++ (increment) -- (decrement) += (compound addition) -= (compound subtraction) \*= (compound multiplication) /= (compound division) &= (compound bitwise and) |= (compound bitwise or)**

**The**

**constant**

**Constants Predefined constants**

**BOOL true false**

**9. User Operation Example**