blynkkk / blynk-server

<> Code | Issues 10 | Pull requests 3 | Actions | Projects | Wiki | Security | Insights

master ▾ | Go to file | Add file ▾ | Code ▾

doom369 Merge pull request #1404 from JamesFromIT/patch-1 … 12 days ago 4,895

client	[maven-release-plugin] prepare for next development ite…	last month
docs	image update	3 years ago
integration-tests	version up	last month
scripts	version up	2 years ago
server	version up	last month
.gitignore	#556 ignore new temp files	4 years ago
.travis.yml	possible build fix with java 10	3 years ago
CODE_OF_CONDUCT.md	Create CODE_OF_CONDUCT.md	4 years ago
CONTRIBUTING.md	Create CONTRIBUTING.md	4 years ago
README.md	version up	last month
apiary.apib	Spelling correction	12 days ago
checkstyle.xml	#917 removed generic web sockets login handler	3 years ago
license.txt	GNU license.	5 years ago

About

Blynk is an Internet of Things Platform aimed to simplify building mobile and web applications for the Internet of Things. Easily connect 400+ hardware models like Arduino, ESP8266, ESP32, Raspberry Pi and similar MCUs and drag-n-drop IOT mobile apps for iOS and Android in 5 minutes

www.blynk.cc/

#java #http #home-automation #mqtt #raspberry-pi #iot #arduino #esp8266 #ota #hardware #netty #rest-api #esp32 #websockets #internet-of-things #particle-photon #iot-platform #ssl-certificate #blynk-server #blynk

Readme

new_server_install.md	version up	last month
pom.xml	[maven-release-plugin] prepare for next development ite…	last month

What is Blynk?

Blynk is a platform with iOS and Android apps to control Arduino, ESP8266, Raspberry Pi and the likes over the Internet.
You can easily build graphic interfaces for all your projects by simply dragging and dropping widgets. If you need more information, please follow these links:

- Blynk site
- Blynk docs
- Blynk community
- Blynk Examples generator
- Facebook
- Twitter
- App Store
- Google Play
- Blynk library
- Kickstarter

GPL-3.0 License

Releases 189

Fixes, improvements Latest
on 28 Apr

+ 188 releases

Packages

No packages published

Contributors 21

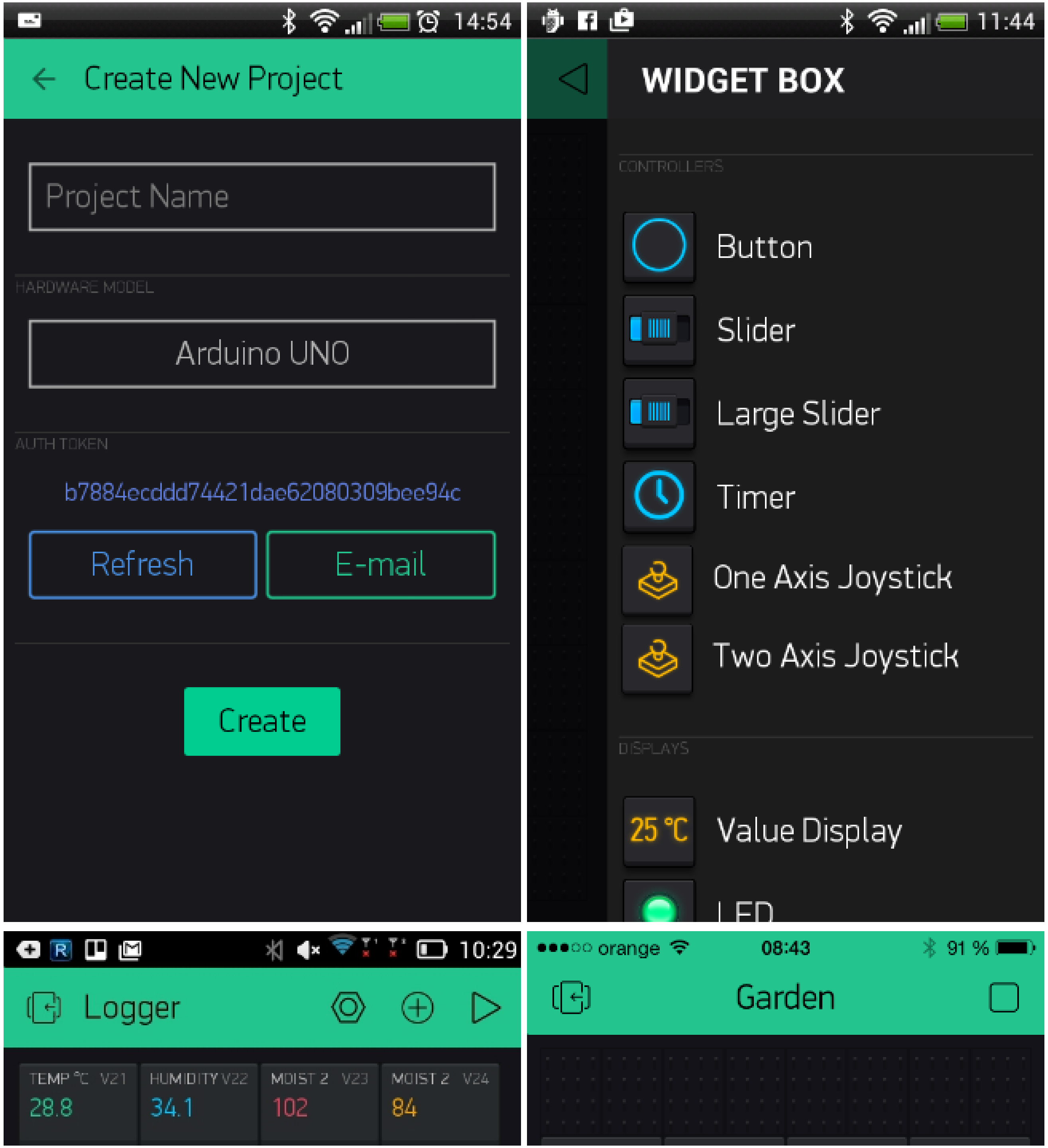
+ 10 contributors

Languages

● Java 97.3% ● HTML 1.5%
● JavaScript 0.6%

● **API Blueprint** 0.4%

● **Shell** 0.1% ● **Dockerfile** 0.1%



14:54

Create New Project

Project Name

HARDWARE MODEL

Arduino UNO

AUTH TOKEN

b7884ecddd74421dae62080309bee94c

Refresh

E-mail

Create

11:44

WIDGET BOX

CONTROLLERS

Button

Slider

Large Slider

Timer

One Axis Joystick

Two Axis Joystick

DISPLAYS

25 ℃ Value Display

LED

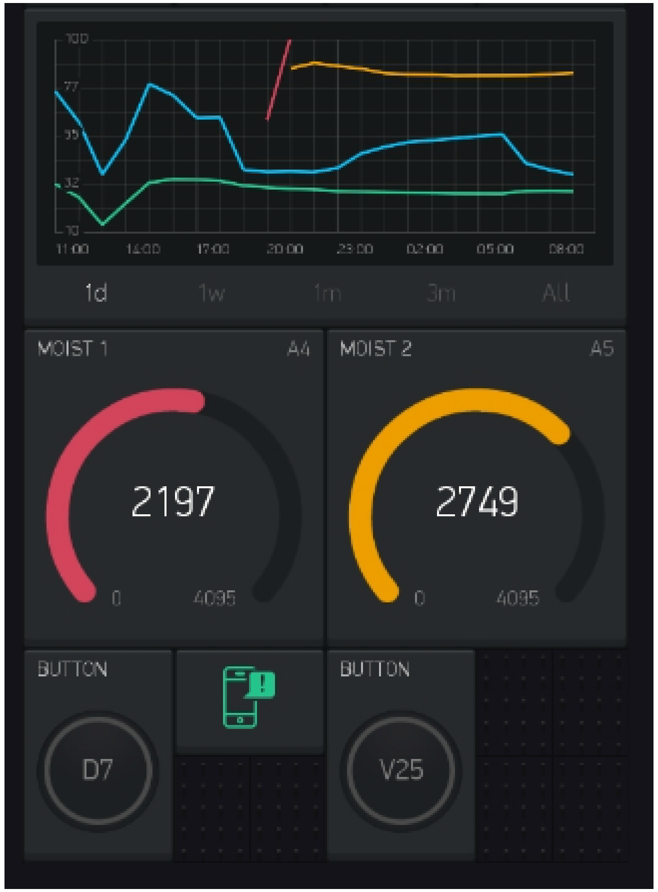
10:29

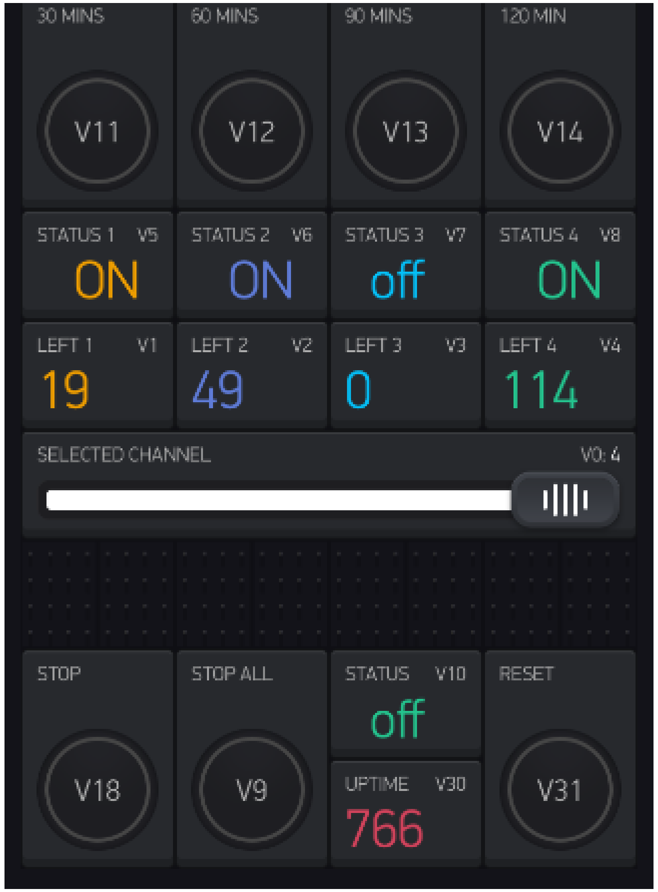
Logger

TEMP ℃ V21	HUMIDITY V22	MOIST 2 V23	MOIST 2 V24
28.8	34.1	102	84

●●●○○ orange 08:43 91 %

Garden


100
77
55
32
10
11:00
14:00
17:00
20:00
23:00
02:00
05:00
08:00
1d
1w
1m
3m
All
MOIST 1 A4
2197
0
4095
MOIST 2 A5
2749
0
4095
BUTTON
D7
BUTTON
V25


30 MINS
V11
60 MINS
V12
90 MINS
V13
120 MIN
V14
STATUS 1 V5
ON
STATUS 2 V6
ON
STATUS 3 V7
off
STATUS 4 V8
ON
LEFT 1 V1
19
LEFT 2 V2
49
LEFT 3 V3
0
LEFT 4 V4
114
SELECTED CHANNEL V0: 4
STOP
V18
STOP ALL
V9
STATUS V10
off
UPTIME V30
766
RESET
V31

Content

GETTING STARTED

Blynk server

Blynk Server is an Open-Source Netty based Java server, responsible for forwarding messages between Blynk mobile application and various microcontroller boards and SBCs (i.e. Arduino, Raspberry Pi. etc).

Download latest server build here.

release v0.41.16 downloads 258k build failing

Requirements

- Java 8/11 required (OpenJDK, Oracle)
- Any OS that can run java
- At least 30 MB of RAM (could be less with tuning)
- Open ports 9443 (for app and hardware with ssl), 8080 (for hardware without ssl)

Ubuntu java installation instruction.

For Windows download Java here and install.

Quick local server setup

- Make sure you are using Java 11

  ```
  java -version
  Output: java version "11"
  ```

- Run the server on default 'hardware port 8080' and default 'application port 9443' (SSL port)

  ```
  java -jar server-0.41.16.jar -dataFolder /path
  ```

That's it!

NOTE: `/path` should be real existing path to folder where you want to store all your data.

- As an output you should see something like that:

  ```
  Blynk Server successfully started.
  All server output is stored in current folder in 'logs/blynk.log' file.
  ```

Enabling mail on Local server

To enable mail notifications on Local server you need to provide your own mail credentials. Create file `mail.properties` within same folder where `server.jar` is. Mail properties:

```
    mail.smtp.auth=true
    mail.smtp.starttls.enable=true
    mail.smtp.host=smtp.gmail.com
    mail.smtp.port=587
    mail.smtp.username=YOUR_EMAIL_HERE
    mail.smtp.password=YOUR_EMAIL_PASS_HERE
```

Find example here.

WARNING : only gmail accounts are allowed.

NOTE : you'll need to setup Gmail to allow less secured applications. Go here and then click "Allow less secure apps".

Quick local server setup on Raspberry PI

- Login to Raspberry Pi via ssh;
- Install java 8:

```
  sudo apt install openjdk-8-jdk openjdk-8-jre
```

- Make sure you are using Java 8

```
  java -version
  Output: java version "1.8"
```

- Download Blynk server jar file (or manually copy it to Raspberry Pi via ssh and scp command):

```
  wget "https://github.com/blynkkk/blynk-server/releases/download/v0.41.16/server-
0.41.16-java8.jar"
```

- Run the server on default 'hardware port 8080' and default 'application port 9443' (SSL port)

```
  java -jar server-0.41.15-java8.jar -dataFolder /home/pi/Blynk
```

That's it!

- As output you will see something like that:

```
  Blynk Server successfully started.
  All server output is stored in current folder in 'logs/blynk.log' file.
```

Docker container setup

Quick Launch

- Install Docker
- Run Docker container

```
docker run -p 8080:8080 -p 9443:9443 mpherg/blynk-server
```

Quick Launch on Raspberry Pi

- Install Docker
- Run Docker container

```
docker run -p 8080:8080 -p 9443:9443 linuxkonsult/rasbian-blynk
```

Full customisation

- Check README in docker folder

Enabling server auto restart on unix-like systems

- To enable server auto restart find /etc/rc.local file and add:

```
java -jar /home/pi/server-0.41.16-java8.jar -dataFolder /home/pi/Blynk &
```

- Or if the approach above doesn't work, execute

```
crontab -e
```

add the following line

```
@reboot java -jar /home/pi/server-0.41.16-java8.jar -dataFolder /home/pi/Blynk &
```

save and exit.

Enabling server auto restart on Windows

- Create bat file:

  ```
  start-blynk.bat
  ```

- Put in it one line:

  ```
  java -jar server-0.41.16.jar -dataFolder /home/pi/Blynk
  ```

- Put bat file to windows startup folder

You can also use this script to run server.

Update instruction for unix-like systems

IMPORTANT Server should be always updated before you update Blynk App. To update your server to a newer version you would need to kill old process and start a new one.

- Find process id of Blynk server

```
ps -aux | grep java
```

- You should see something like that

```
  username   10539  1.0 12.1 3325808 428948 pts/76 Sl   Jan22   9:11 java -jar
server-0.41.16.jar
```

- Kill the old process

```
kill 10539
```

10539 - blynk server process id from command output above.

- Start new server as usual

After this steps you can update Blynk app. Server version downgrade is not supported.

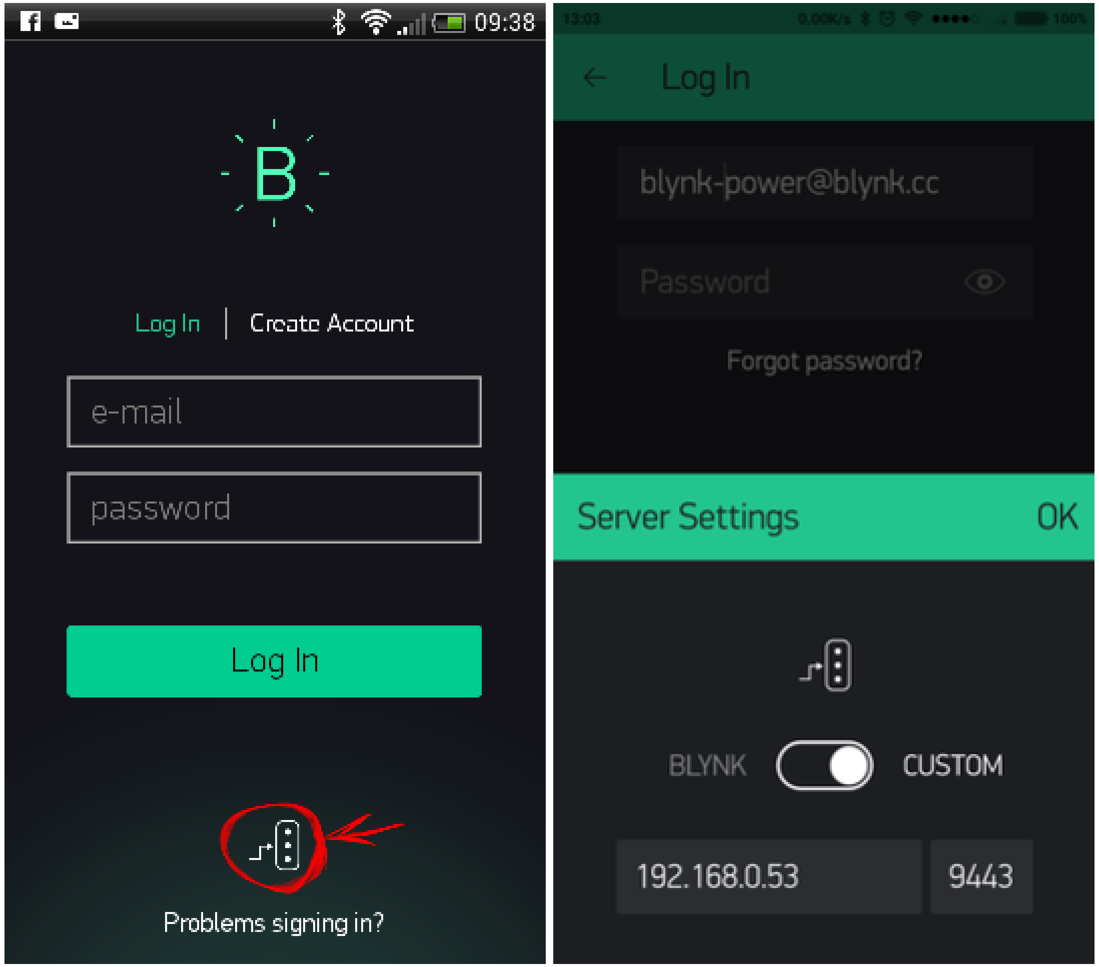
WARNING! Please **do not** revert your server to lower versions. You may loose all of your data.

Update instruction for Windows

- Open Task Manager;
- Find Java process;
- Stop process;
- Start new server as usual

App and sketch changes

- Specify custom server path in your application



blynk-power@blynk.cc

192.168.0.53 9443

- Change your ethernet sketch from

```
Blynk.begin(auth);
```

to

```
Blynk.begin(auth, "your_host", 8080);
```

or to

```
Blynk.begin(auth, IPAddress(xxx,xxx,xxx,xxx), 8080);
```

- Change your WIFI sketch from

```
Blynk.begin(auth, SSID, pass));
```

to

```
Blynk.begin(auth, SSID, pass, "your_host", 8080);
```

or to

```
Blynk.begin(auth, SSID, pass, IPAddress(XXX,XXX,XXX,XXX), 8080);
```

- Change your rasp PI javascript from

```
var blynk = new Blynk.Blynk(AUTH, options = {connector : new Blynk.TcpClient()});
```

to

```
var blynk = new Blynk.Blynk(AUTH, options= {addr:"xxx.xxx.xxx.xxx", port:8080});
```

- or in case of USB when running blynk-ser.sh provide '-s' option with address of your local server

```
  ./blynk-ser.sh -s you_host_or_IP
```

IMPORTANT Blynk is being constantly developed. Mobile apps and server are updated often. To avoid problems during updates either turn off auto-update for Blynk app, or update both local server and blynk app at same time to avoid possible migration issues.

IMPORTANT Blynk local server is different from Blynk Cloud server. They are not related at all. You have to create new account when using Blynk local server.

Advanced local server setup

For more flexibility you can extend server with more options by creating `server.properties` file in same folder as `server.jar`. Example could be found here. You could also specify any path to `server.properties` file via command line argument `-serverConfig`. You can do the same with `mail.properties` via `-mailConfig` and `sms.properties` via `-smsConfig`.

For example:

```
java -jar server-0.41.16-java8.jar -dataFolder /home/pi/Blynk -serverConfig
/home/pi/someFolder/server.properties
```

Available server options:

- Blynk app, https, web sockets, admin port

```
https.port=9443
```

- Http, hardware and web sockets port

```
http.port=8080
```

- For simplicity Blynk already provides server jar with built in SSL certificates, so you have working server out of the box via SSL/TLS sockets. But as certificate and it's private key are in public this is totally not secure. So in order to fix that you need to provide your own certificates. And change below properties with path to your cert. and private key and it's password. See how to generate self-signed certificates here

```
#points to cert and key that placed in same folder as running jar.

server.ssl.cert=./server_embedded.crt
server.ssl.key=./server_embedded.pem
server.ssl.key.pass=pupkin123
```

Note: if you use Let's Encrypt certificates you'll have to add `#define BLYNK_SSL_USE_LETSENCRYPT` before `#include <BlynkSimpleEsp8266_SSL.h>` in the Arduino Sketch for your hardware.

- User profiles folder. Folder in which all users profiles will be stored. By default System.getProperty("java.io.tmpdir")/blynk used. Will be created if not exists

```
data.folder=/tmp/blynk
```

- Folder for all application logs. Will be created if it doesn't exist. "." is dir from which you are running script.

```
logs.folder=./logs
```

- Log debug level. Possible values: trace|debug|info|error. Defines how precise logging will be. From left to right -> maximum logging to minimum

```
log.level=trace
```

- Maximum allowed number of user dashboards.

```
user.dashboard.max.limit=100
```

- 100 Req/sec rate limit per user. You also may want to extend this limit on hardware side.

```
user.message.quota.limit=100
```

- this setting defines how often you can send mail/tweet/push or any other notification. Specified in seconds

```
notifications.frequency.user.quota.limit=60
```

- Maximum allowed user profile size. In Kb's.

```
user.profile.max.size=128
```

- Number of strings to store in terminal widget (terminal history data)

```
terminal.strings.pool.size=25
```

- Maximum allowed number of notification queue. Queue responsible for processing email, pushes, twits sending. Because of performance issue - those queue is processed in separate thread, this is required due to blocking nature of all above operations. Usually limit shouldn't be reached

```
notifications.queue.limit=5000
```

- Number of threads for performing blocking operations - push, twits, emails, db queries. Recommended to hold this value low unless you have to perform a lot of blocking operations.

```
blocking.processor.thread.pool.limit=6
```

- Period for flushing all user DB to disk. In millis

```
profile.save.worker.period=60000
```

- Specifies maximum period of time when hardware socket could be idle. After which socket will be closed due to non activity. In seconds. Leave it empty for infinity timeout

```
hard.socket.idle.timeout=15
```

- Mostly required for local servers setup in case user want to log raw data in CSV format. See [raw data] (#raw-data-storage) section for more info.

```
enable.raw.data.store=true
```

- Url for opening admin page. Must start from "/". For "/admin" url path will look like that "https://127.0.0.1:9443/admin".

```
admin.rootPath=/admin
```

- Comma separated list of administrator IPs. Allow access to admin UI only for those IPs. You may set it for 0.0.0.0/0 to allow access for all. You may use CIDR notation. For instance, 192.168.0.53/24.

```
allowed.administrator.ips=0.0.0.0/0
```

- Default admin name and password. Will be created on initial server start

```
admin.email=admin@blynk.cc
admin.pass=admin
```

- Host for reset password redirect and certificate generation. By default current server IP is taken from "eth" network interface. Could be replaced with more friendly hostname. It is recommended to override this property with your server IP to avoid possible problems of host resolving.

```
server.host=blynk-cloud.com
```

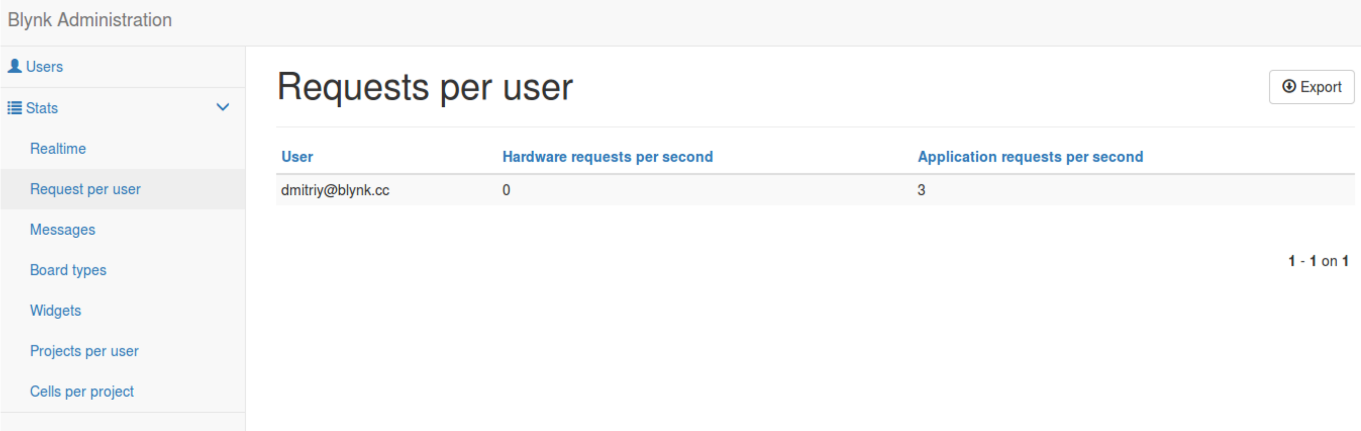
- Email used for certificate registration, could be omitted in case you already specified it in mail.properties.

```
contact.email=pupkin@gmail.com
```

Administration UI

Blynk server provides administration panel where you can monitor your server. It is accessible at this URL:

```
https://your_ip:9443/admin
```



Requests per user

User	Hardware requests per second	Application requests per second
dmitriy@blynk.cc	0	3

WARNING Please change default admin password and name right after login to admin page. **THIS IS SECURITY MEASURE**.

WARNING Default `allowed.administrator.ips` setting allows access for everyone. In other words, administration page available from any other computer. Please restrict access to it via property `allowed.administrator.ips`.

Turn off chrome https warning on localhost

- Paste in chrome

  ```
  chrome://flags/#allow-insecure-localhost
  ```

- You should see highlighted text saying: "Allow invalid certificates for resources loaded from localhost". Click enable.

HTTP/S RESTful

Blynk HTTP/S RESTful API allows to easily read and write values to/from Pins in Blynk apps and Hardware. Http API description could be found here.

Enabling sms on local server

To enable SMS notifications on Local Server you need to provide credentials for SMS gateway (currently Blynk server supports only 1 provider - Nexmo. You need to create file `sms.properties` within same folder where server.jar is.

```
    nexmo.api.key=
    nexmo.api.secret=
```

And fill in the above properties with the credentials you'll get from Nexmo. (Account -> Settings -> API settings). You can also send SMS over email if your cell provider supports that. See discussion for more details.

Enabling raw data storage

By default raw data storage is disabled (as it consumes disk space a lot). When you enable it, every `Blynk.virtualWrite` command will be saved to DB. You will need to install PostgreSQL Database (**minimum required version is 9.5**) to enable this functionality:

1. Enabling raw data on server

Enable raw data in `server.properties` :

```
    enable.db=true
    enable.raw.db.data.store=true
```

2. Install PostgreSQL. Option A

```
    sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" >> /etc/apt/sources.list.d/pgdg.list'
    wget -q https://www.postgresql.org/media/keys/ACCC4CF8.asc -O - | sudo apt-key add -

    sudo apt-get update
    sudo apt-get install postgresql postgresql-contrib
```

2. Install PostgreSQL. Option B

```
    sudo apt-get update
    apt-get --no-install-recommends install postgresql-9.6 postgresql-contrib-9.6
```

3. Download Blynk DB script

```
    wget https://raw.githubusercontent.com/blynkkk/blynk-server/master/server/core/src/main/resources/create_schema.sql
    wget https://raw.githubusercontent.com/blynkkk/blynk-
server/master/server/core/src/main/resources/reporting_schema.sql
```

4. Move create_schema.sql and reporting_schema.sql to temp folder (to avoid permission problems)

```
    mv create_schema.sql /tmp
    mv reporting_schema.sql /tmp
```

Result:

```
    /tmp/create_schema.sql
    /tmp/reporting_schema.sql
```

Copy it to clipboard from your console.

5. Connect to PostgreSQL

```
    sudo su - postgres
    psql
```

6. Create Blynk DB and Reporting DB, test user and tables

```
    \i /tmp/create_schema.sql
    \i /tmp/reporting_schema.sql
```

`/tmp/create_schema.sql` - is path from step 4.

You should see next output:

```
    postgres=# \i /tmp/create_schema.sql
    CREATE DATABASE
    You are now connected to database "blynk" as user "postgres".
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE TABLE
    CREATE ROLE
    GRANT
    GRANT
```

Quit

```
    \q
```

Now start your server and you should see next text in `postgres.log` file :

```
    2017-03-02 16:17:18.367 - DB url : jdbc:postgresql://localhost:5432/blynk?
tcpKeepAlive=true&socketTimeout=150
    2017-03-02 16:17:18.367 - DB user : test
    2017-03-02 16:17:18.367 - Connecting to DB...
    2017-03-02 16:17:18.455 - Connected to database successfully.
```

WARNING: Raw data may consume your disk space very quickly!

CSV data format

Data format is:

```
value,timestamp,deviceId
```

For example:

```
10,1438022081332,0
```

Where `10` - value of pin. `1438022081332` - the difference, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC. To display the date/time in excel you may use formula:

```
=((COLUMN/(60*60*24)/1000+25569))
```

`0` - device id

Automatic Let's Encrypt certificates generation

Latest Blynk server has super cool feature - automatic Let's Encrypt certificates generation. However, it has few requirements:

- Add `server.host` property in `server.properties` file. For example :

  ```
  server.host=myhost.com
  ```

IP is not supported, this is the limitation of Let's Encrypt. Also have in mind that `myhost.com` should be resolved by public DNS severs.

- Add `contact.email` property in `server.properties` . For example :

  ```
  contact.email=test@gmail.com
  ```

- You need to start server on port 80 (requires root or admin rights) or make port forwarding to default Blynk HTTP port - 8080.

That's it! Run server as regular and certificates will be generated automatically.

Manual Let's Encrypt SSL/TLS Certificates

- First install certbot on your server (machine where you going to run Blynk Server)

  ```
  wget https://dl.eff.org/certbot-auto
  chmod a+x certbot-auto
  ```

- Generate and verify certificates (your server should be connected to internet and have open 80/443 ports)

  ```
  ./certbot-auto certonly --agree-tos --email YOUR_EMAIL --standalone -d YOUR_HOST
  ```

For example

```
    ./certbot-auto certonly --agree-tos --email pupkin@blynk.cc --standalone -d
blynk.cc
```

- Then add to your `server.properties` file (in folder with server.jar)

```
server.ssl.cert=/etc/letsencrypt/live/YOUR_HOST/fullchain.pem
server.ssl.key=/etc/letsencrypt/live/YOUR_HOST/privkey.pem
server.ssl.key.pass=
```

Generate own SSL certificates

- Generate self-signed certificate and key

```
  openssl req -x509 -nodes -days 1825 -newkey rsa:2048 -keyout server.key -out
server.crt
```

- Convert server.key to PKCS#8 private key file in PEM format

```
openssl pkcs8 -topk8 -v1 PBE-SHA1-2DES -in server.key -out server.enc.key
```

If you connect hardware with USB script you have to provide an option '-s' pointing to "common name" (hostname) you did specified during certificate generation.

As an output you'll retrieve server.crt and server.pem files that you need to provide for server.ssl properties.

Install java for Ubuntu

```
sudo add-apt-repository ppa:openjdk-r/ppa \
&& sudo apt-get update -q \
&& sudo apt install -y openjdk-11-jdk
```

or if above doesn't work:

```
sudo apt-add-repository ppa:webupd8team/java
sudo apt-get update
sudo apt-get install oracle-java8-installer
```

Port forwarding for HTTP/S API

```
sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8080
sudo iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-port 9443
```

Enabling QR generation on server

```
sudo apt-get install libxrender1
```

Behind wifi router

If you want to run Blynk server behind WiFi-router and want it to be accessible from the Internet, you have to add port-forwarding rule on your router. This is required in order to forward all of the requests that come to the router within the local network to Blynk server.

How to build

Blynk has a bunch of integration tests that require DB, so you have to skip tests during build.

```
mvn clean install -Dmaven.test.skip=true
```

How Blynk Works?

When hardware connects to Blynk cloud it opens either keep-alive ssl/tls connection on port 443 (9443 for local servers) or keep-alive plain tcp/ip connection on port 8080. Blynk app opens mutual ssl/tls connection to Blynk Cloud on port 443 (9443 for local servers). Blynk Cloud is responsible for forwarding messages between hardware and app. In both (app and hardware) connections Blynk uses own binary protocol described below.

Blynk protocol

Hardware side protocol

Blynk transfers binary messages between the server and the hardware with the following structure:

Command	Message Id	Length/Status	Body
1 byte	2 bytes	2 bytes	Variable

Command and Status definitions: BlynkProtocolDefs.h

Mobile app side protocol

Blynk transfers binary messages between the server and mobile app with the following structure:

Command	Message Id	Length/Status	Body
1 byte	2 bytes	4 bytes	Variable

Websockets web side protocol

Blynk transfers binary messages between the server and websockets (for web) with the following structure:

Websocket header	Command	Message Id	Body
	1 byte	2 bytes	Variable

When command code == 0, than message structure is next:

Websocket header	Command	Message Id	Response code
	1 byte	2 bytes	4 bytes

Possible response codes. Possible command codes

Message Id and Length are big endian. Body has a command-specific format.

Licensing

GNU GPL license