Lamonte Networking docs

# 

[0. Prerequisites](#h.vt58ki79kmay)

[0.0 Help to improve this document](#h.mxhj1de9z6od)

[0.1 Prepare console](#h.8g3jr527iaps)

[1 Auth](#h.iuhtigk8x4l3)

[1.1 Obtain token](#h.r9wr758xzo9g)

[1.2 Prepare console](#h.ei9vyvelqo3z)

[2 Bags](#h.wgzsqoniywbq)

[2.1 List bags (GET)](#h.ss8xhu90fk02)

[2.2 Create bag (POST)](#h.gh0tpncygx91)

[2.2 Get bag (GET)](#h.i2pg3sgnf588)

[2.3 Update bag (PUT)](#h.b4o0uvk5zt4t)

[2.4 Remove bag (DELETE)](#h.iis60lf4wtsr)

[3. Contacts](#h.sdaocirk9ny1)

[3.1 List contacts (GET)](#h.nunjk01pkqxn)

[3.2 Create contact (POST)](#h.x6rcgjtvtu9b)

[3.3 Get contact (GET)](#h.emhtgbt5zix4)

[3.4 Remove contact (DELETE)](#h.swj98fziftdu)

[4. Realtime server (Socket.IO)](#h.w80tno25hag4)

[4.0 Messages](#h.st8gngc5yiue)

[4.1 Connection](#h.dztto6f5hn36)

[4.2 Session](#h.6kj0i1dt4pm)

[4.3 Echo](#h.ika9lzgmsrmd)

[4.4 Movement](#h.hhxph3tlfece)

[4.5 Away (Stolen)](#h.8uc3ktlq93ht)

[4.6 Socket client delegate protocol (interface)](#h.j2mn5k7n2vlu)

[4.7 Full code of SocketClient](#h.b585m6b7kd5k)

[5. Device Data Entries](#h.njnbd47fya64)

[5.1 List device data entries (GET)](#h.xownm2io4pkv)

[5.2 Create device data entry (POST)](#h.ze56l9880mug)

# 

# 

# **0. Prerequisites**

Install `httpie` app (<https://github.com/jkbrzt/httpie> , for Mac `brew install httpie`, it’s better than curl so I use it for samples, READ THE DOCS FOR HTTPIE, notice that == sign is GET param and = is POST param

My demonstration showing idea of the app <https://yadi.sk/i/rRyqxjG_ivzfh> (feel free skip boring parts :-)

Lamonte administration is available at <http://lamonte-london.com/api/admin/>

For API access please refer to <http://lamonte-london.com/api/v1/>

Web API interface is self-documented, use HTML form at the bottom of the screen to test requests

**BASE URL IS** [**http://lamonte-london.com/api**](http://lamonte-london.com/api) **APPEND REQUIRED PARTS AS FOLLOWS, `/v1/` IS NOT USED FOR SOME CALLS**

**TRAILING SLASH IS REQUIRED, OTHERWISE YOU’LL GET `HTTP/1.1 301 MOVED PERMANENTLY`**

**PLEASE USE CLASSES, TYPE CHECKS AND TESTS, NOT DICTIONARIES AND ARRAYS IN YOUR CODE**

## 0.0 Help to improve this document

I pay huge attention to correctness of documentation and code, so HTTP status codes, parameter types, URL parts and so on should be correct. Please let me know if there are any issues or ways to improve documentation or code

## 0.1 Prepare console

All samples tested on my Mac in Terminal. Linux is fine too. On Windows you can use `cygwin` but Powershell would be fine too, just find a way to reproduce this behaviour (I use env variables for brevity):

export S=[**http://lamonte-london.com/api**](http://lamonte-london.com/api) **# set S env variable**

echo $S **# test**

# 1 Auth

## 1.1 Obtain token

**Administration page**

<http://lamonte-london.com/api/admin/authtoken/token/>

**api-token-auth/**

http $S/api-token-auth/ username=nmtitov@nmtitov.com password=123

output:

HTTP/1.1 200 OK

{

"token": "04ea82443d876c282e5ffc4211cba06af99e5a4b"

}

token should be used for all requests which require auth

Pass token as header “Authorization”: “Token 04ea82443d876c282e5ffc4211cba06af99e5a4b”

## 1.2 Prepare console

http $S/v1/users/ **# ensure that auth is required**

http $S/v1/users/ Authorization:"Token 04ea82443d876c282e5ffc4211cba06af99e5a4b" **# ensure token is correct**

export T=04ea82443d876c282e5ffc4211cba06af99e5a4b **# set T env variable**

http $S/v1/users/ Authorization:"Token $T" **# test**

output

HTTP/1.1 200 OK

{

"count": 4,

"next": null,

"previous": null,

"results": [

{

"email": "2@2.com",

"id": 4,

"lat": null,

"lon": null,

"url": "http://lamonte-london.com/api/v1/users/4/"

},

…

# 2 Bags

## 2.1 List bags (GET)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/bag/**](http://lamonte-london.com/api/admin/lamonte_core/bag/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/**](http://lamonte-london.com/api/v1/bags/)

**v1/bags/**

http $S/v1/bags/ Authorization:"Token $T"

output

HTTP/1.1 200 OK

{

"count": 3,

"next": null,

"previous": null,

"results": [

{

"battery": 1.0,

"charging": false,

"code": "nikita\_1",

"contacts": [

{

"bag": 10,

"e164Phone": "+79110739371",

"formattedPhone": "+79110739371",

"id": 1,

"iso": "ru",

"name": "Nikita",

"phone": "+79110739371"

}

],

...

## 2.2 Create bag (POST)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/bag/add/**](http://lamonte-london.com/api/admin/lamonte_core/bag/add/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/**](http://lamonte-london.com/api/v1/bags/)

**params**

name

code (unique)

geo\_fence

tracking

**v1/bags/**

http $S/v1/bags/ Authorization:"Token $T" name="My New Bag" code=12345xyz geo\_fence=125 tracking=true

output (data of new bag, for simpler UI update, please use classes and type checking instead of hashes and arrays)

HTTP/1.1 201 CREATED

{

"battery": 1.0,

"charging": false,

"code": "12345xyz",

"contacts": [],

"distance": null,

"geo\_fence": 125.0,

"id": 14,

"image": null,

"lat": null,

"lon": null,

"name": "My New Bag",

"nearby": false,

"tracking": true

}

## 2.2 Get bag (GET)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/bag/10/**](http://lamonte-london.com/api/admin/lamonte_core/bag/10/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/10/**](http://lamonte-london.com/api/v1/bags/10/)

**v1/bags/10/ # use real id**

http $S/v1/bags/10/ Authorization:"Token $T"

output (data of existing bag, for simpler UI update, please use classes and type checking instead of hashes and arrays)

HTTP/1.1 200 OK

{

"battery": 1.0,

"charging": false,

"code": "nikita\_1",

"contacts": [

{

"bag": 10,

"e164Phone": "+11012340001",

"formattedPhone": "+11012340001",

"id": 2,

"iso": "US",

"name": "My bodyguard",

"phone": "+11012340001"

},

{

"bag": 10,

"e164Phone": "+79110739371",

"formattedPhone": "+79110739371",

"id": 1,

"iso": "ru",

"name": "Nikita",

"phone": "+79110739371"

}

],

"distance": null,

"geo\_fence": 10.0,

"id": 10,

"image": null,

"lat": null,

"lon": null,

"name": "My bag",

"nearby": false,

"tracking": true

}

## 2.3 Update bag (PUT)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/bag/10/**](http://lamonte-london.com/api/admin/lamonte_core/bag/10/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/10/**](http://lamonte-london.com/api/v1/bags/10/)

**v1/bags/14/ # use real id**

http PUT $S/v1/bags/14/ Authorization:"Token $T" name="My Old Bag" code=12345xyz geo\_fence=10 tracking=false

output (data of new bag, for simpler UI update, please use classes and type checking instead of hashes and arrays)

HTTP/1.1 200 OK

{

"battery": 1.0,

"charging": false,

"code": "12345xyz",

"contacts": [],

"distance": null,

"geo\_fence": 10.0,

"id": 14,

"image": null,

"lat": null,

"lon": null,

"name": "My Old Bag",

"nearby": false,

"tracking": false

}

## 2.4 Remove bag (DELETE)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/bag/10/**](http://lamonte-london.com/api/admin/lamonte_core/bag/10/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/10/**](http://lamonte-london.com/api/v1/bags/10/)

**v1/bags/14/ # carefully use real id**

http DELETE $S/v1/bags/14/ Authorization:"Token $T"

output

HTTP/1.1 204 NO CONTENT

# 3. Contacts

Contacts assigned for each bag. When bag is out of geo fence, contacts would receive sms with bag’s last seen location or coordinate

## 3.1 List contacts (GET)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/10/contacts/**](http://lamonte-london.com/api/v1/bags/10/contacts/)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/contact/**](http://lamonte-london.com/api/admin/lamonte_core/contact/)

**OR**

[**http://lamonte-london.com/api/admin/lamonte\_core/bag/10/**](http://lamonte-london.com/api/admin/lamonte_core/bag/10/) **(contacts embedded in bag form)**

**v1/bags/10/contacts/ # use real id**

http $S/v1/bags/10/contacts/ Authorization:"Token $T"

output

HTTP/1.1 200 OK

{

"count": 1,

"next": null,

"previous": null,

"results": [

{

"bag": 10,

"e164Phone": "+79110739371",

"formattedPhone": "+79110739371",

"id": 1,

"iso": "ru",

"name": "Nikita",

"phone": "+79110739371"

}

...

## 

## 

## 3.2 Create contact (POST)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/contact/**](http://lamonte-london.com/api/admin/lamonte_core/contact/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/10/contacts/**](http://lamonte-london.com/api/v1/bags/10/contacts/)

**USE THIS LIBRARY FOR PHONE NUMBERS HANDLING** [**https://github.com/googlei18n/libphonenumber**](https://github.com/googlei18n/libphonenumber)

**Difference between phone, e164Phone and formatted phone is explained on page linked above. Numbers should be valid.**

**params**

name

iso (<https://ru.wikipedia.org/wiki/ISO_3166-2>)

phone

e164phone

formattedPhone

**v1/bags/10/contacts/**

http $S/v1/bags/10/contacts/ Authorization:"Token $T" name="My bodyguard" iso="US" phone="+11012340001" e164Phone="+11012340001" formattedPhone="+11012340001"

output (data of new contact, for simpler UI update, please use classes and type checking instead of hashes and arrays)

HTTP/1.1 201 CREATED

{

"bag": 10,

"e164Phone": "+11012340001",

"formattedPhone": "+11012340001",

"id": 2,

"iso": "US",

"name": "My bodyguard",

"phone": "+11012340001"

}

## 3.3 Get contact (GET)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/contact/1/**](http://lamonte-london.com/api/admin/lamonte_core/contact/1/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/10/contacts/1/**](http://lamonte-london.com/api/v1/bags/10/contacts/1/)

**v1/bags/10/contacts/2/ # use real id**

http $S/v1/bags/10/contacts/2/ Authorization:"Token $T"

output (data of existing bag, for simpler UI update, please use classes and type checking instead of hashes and arrays)

HTTP/1.1 200 OK

{

"bag": 10,

"e164Phone": "+11012340001",

"formattedPhone": "+11012340001",

"id": 2,

"iso": "US",

"name": "My bodyguard",

"phone": "+11012340001"

},

## 

## 

## 3.4 Remove contact (DELETE)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/contact/1/**](http://lamonte-london.com/api/admin/lamonte_core/contact/1/)

**Web API page**

[**http://lamonte-london.com/api/v1/bags/10/contacts/1/**](http://lamonte-london.com/api/v1/bags/10/contacts/1/)

**USE THIS LIBRARY FOR PHONE NUMBERS HANDLING** [**https://github.com/googlei18n/libphonenumber**](https://github.com/googlei18n/libphonenumber)

**Difference between phone, e164Phone and formatted phone is explained on page linked above. Numbers should be valid.**

**params**

name

iso (<https://ru.wikipedia.org/wiki/ISO_3166-2>)

phone

e164phone

formattedPhone

**v1/bags/10/contacts/2/ # carefully use real id**

http $S/v1/bags/10/contacts/2/ Authorization:"Token $T"

output

HTTP/1.1 204 NO CONTENT

# 4. Realtime server (Socket.IO)

I use Socket.IO libraries for realtime server and they add their own paths to base url, don’t forget about that and don’t try to use raw websocket libraries. Please use officially supported Socket.IO libraries.

**BASE URL IS** [lamonte-london.com](http://lamonte-london.com)

Examples are given in Objective-C code

## 4.0 Messages

static NSString \* const kECHO = @"echo";

static NSString \* const kBAG\_MOVE = @"bag\_move";

static NSString \* const kBAG\_AWAY = @"bag\_away";

## 4.1 Connection

Socket.IO is picky about requests and you’ll want to make sure that you are able to establish connection successfully. Once you get token from REST API, connect to Socket.IO using token as `key` parameter

- (void)connect {

NSDictionary \*params = @{

@"key": App().token,

};

[self.socket connectWithParams:params];

}

## 4.2 Session

- (void)addSessionHandlers {

@weakify(self);

[self.socket on:@"error" callback:^(NSArray\* data, void (^ack)(NSArray\*)) {

NSLog(@"App server error: %@", data.firstObject);

}];

[self.socket on:@"connect" callback:^(NSArray\* data, void (^ack)(NSArray\*)) {

@strongify(self);

[self.delegate didConnect];

}];

[self.socket on:@"disconnect" callback:^(NSArray \*data, void (^ack)(NSArray\*)) {

@strongify(self);

[self.delegate didDisconnect];

}];

}

## 4.3 Echo

Request

- (void)echo {

[self.socket emit:@"echo" withItems:@[@"Hello?"]];

}

Response format (please use classes and type checks, not dictionaries or hashes):

{

"echo": string",

}

This is how I handle response

- (void)addEchoHandlers {

@weakify(self);

[self.socket on:kECHO callback:^(NSArray \*data, void (^ack)(NSArray\*)) {

@strongify(self);

NSError \*error;

LEchoSocketMessage \*echo = [MTLJSONAdapter modelOfClass:[LEchoSocketMessage class] fromJSONDictionary:data.firstObject error:&error];

if (!echo) {

NSLog(@"Could not parse echo message: %@", error.localizedDescription);

return;

}

[self.delegate didEcho:echo];

}];

}

## 4.4 Movement

Response format (please use classes and type checks, not dictionaries or hashes):

{

"serverID": int,

"lat": double,

"lon": double,

"distance": int,

"battery": double,

"charging": bool

}

This is how I handle response

- (void)addMovementHandlers {

@weakify(self);

[self.socket on:kBAG\_MOVE callback:^(NSArray \*data, void (^ack)(NSArray\*)) {

@strongify(self);

NSError \*error;

NSDictionary \*dict = data.firstObject;

LBagMovementSocketMessage \*movement =

[MTLJSONAdapter modelOfClass:[LBagMovementSocketMessage class] fromJSONDictionary:dict error:&error];

if (!movement) {

NSLog(@"App server error: %@", error.localizedDescription);

return;

}

[self.delegate bagDidMove:movement];

}];

}

## 4.5 Away (Stolen)

Response format (same as movement, please use classes and type checks, not dictionaries or hashes):

{

"serverID": int,

"lat": double,

"lon": double,

"distance": int,

"battery": double,

"charging": bool

}

This is how I handle response

- (void)addAwayHandlers {

@weakify(self);

[self.socket on:kBAG\_AWAY callback:^(NSArray \*data, void (^ack)(NSArray\*)) {

@strongify(self);

NSError \*error;

LBagMovementSocketMessage \*movement = [MTLJSONAdapter modelOfClass:[LBagMovementSocketMessage class] fromJSONDictionary:data.firstObject error:&error];

if (!movement) {

NSLog(@"App server error: %@", error.localizedDescription);

return;

}

[self.delegate bagDidMoveAway:movement];

}];

}

## 4.6 Socket client delegate protocol (interface)

@protocol LSocketClientDelegate <NSObject>

- (void)didReceiveServerError:(id)serverError;

- (void)didConnect;

- (void)didDisconnect;

- (void)didEcho:(LEchoSocketMessage \*)echo;

- (void)bagDidMove:(LBagMovementSocketMessage \*)bag;

- (void)bagDidMoveAway:(LBagMovementSocketMessage \*)bag;

@end

## 4.7 Full code of SocketClient

LSocketClient.h, LSocketClient.m — LSocketClient class

LBagsMasterController.m — shows usage of LSocketClientDelegate

<https://gist.github.com/nmtitov/1a87c5feef38625dec4b>

# 5. Device Data Entries

This API endpoint doesn’t require authentication at the moment

## 5.1 List device data entries (GET)

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/devicedataentry/**](http://lamonte-london.com/api/admin/lamonte_core/devicedataentry/)

**Web API page**

[**http://lamonte-london.com/api/v1/device\_data\_entries/**](http://lamonte-london.com/api/v1/device_data_entries/)

http $S/v1/device\_data\_entries/

output

HTTP/1.1 200 OK

{

"count": 1,

"next": null,

"previous": null,

"results": [

{

"id": 1,

"created": "2016-02-05T12:44:16.418077Z",

"data": "test",

}

...

## 

## 

## 5.2 Create device data entry (POST)

This API endpoint doesn’t require authentication at the moment

**Administration page**

[**http://lamonte-london.com/api/admin/lamonte\_core/devicedataentry/add/**](http://lamonte-london.com/api/admin/lamonte_core/devicedataentry/add/)

**Web API page**

[**http://lamonte-london.com/api/v1/device\_data\_entries/**](http://lamonte-london.com/api/v1/device_data_entries/)

**params**

data:string

**v1/device\_data\_entries/**

http $S/v1/device\_data\_entries/ data="test"

output (data of new contact, for simpler UI update, please use classes and type checking instead of hashes and arrays)

HTTP/1.1 201 CREATED

{

"id": 1,

"created": "2016-02-05T12:44:16.418077Z",

"data": "test",

}