

API KEY	2
BASE URL	2
Models	3
Operation Result Object	3
AUTH	4
auth/login	4
auth/logout	4
auth/me	5
SECURITY	6
security/get-captcha-url	6
USERS	7
/users	7
PROFILE	8
profile/status/{userId}	8
profile/status	8
profile/{userId}	8
profile	9
profile/photo	10
FOLLOWING	12
follow/{userId}	12
follow/{userId}	12
follow/{userId}	12
DIALOGS	
dialogs	
dialogs/{userId}/messages	
dialogs/{userId}/messages	
dialogs/{userId}	
dialogs/messages/{messageId}/viewed	
dialogs/messages/{messageId}/spam	
dialogs/messages/{messageId}	
dialogs/messages/{messageId}/restore	
dialogs/{userId}/messages/new?newerThan={date}	
dialogs/messages/new/count	

API KEY

Get your own API KEY here <https://social-network.samuraijs.com/account>
and add it to http request headers:

Example:

```
headers: {  
  'API-KEY': 'd0ce4cd8-d0d2-4152-99ad-f6e1407cb23f'  
}
```

BASE URL

BASE URL: <https://social-network.samuraijs.com/api/1.0/>

Models

Json Models

Operation Result Object

```
{  
  resultCode: 1  
  messages: ['Something wrong'],  
  data: {}  
}
```

- resultCode (integer) - 0 - OK, other codes - request is invalid
- messages (string[]) - is empty if **resultCode is 0**, contains messages if **resultCode is different from 0**
- data (object) - contains additional information that depends on concrete request

AUTH

Authorization workflow

auth/login

REQUEST

type: **post**

required parameters:

- **email** (string) - valid confirmed user email address, which used during registration
- **password** (string) - valid user password
- **rememberMe** (bool) - if *true*, then session will not be expired after session finishing

partially required parameters:

- **captcha** (string) - should be added if there is **resultCode is 10** in **response** data.

[Read how to get captcha image](#)

RESPONSE

json object

properties:

resultCode (integer) - 0 - OK, 1 - request is invalid, 10 - request is invalid and **captcha** is required

messages (string[]) - is empty if **resultCode is 0**, contains messages if **resultCode is different from 0**

data (object) - is empty if **resultCode is 1 or 10**, if **resultCode is 0** then object contains

* **userId** - logged user id

auth/logout

REQUEST

type: **post**

no parameters

RESPONSE

json object

properties:

resultCode (integer) - 0 - OK, 1 - request is invalid

auth/me

REQUEST

type: **get**

RESPONSE

json object

properties:

resultCode (integer) - 0 - OK, 1 - request is invalid

messages (string[]) - is empty if **resultCode is 0**, contains error messages if **resultCode is different from 0**

data (object) - is empty if **resultCode is not equal 0**, if **resultCode is 0** then object contains

* id - logged user id

* email - logged user email

* login - user login

SECURITY

Security workflow

security/get-captcha-url

REQUEST

type: **get**

RESPONSE

json object

properties:

- **url**: url to display captcha-image

USERS

Users information

/users

Get list of users

REQUEST

type: **get**

optional params:

count - page size (how many items will be returned in response)

page - number of portion of items

term - user name string for searching

RESPONSE

json object

properties:

- **items**: array[] of **User** - lists of available users
- **totalCount**: integer - common amount of registered users

PROFILE

Work with user profile (description, hobbies, work, status, etc...)

profile/status/{userId}

REQUEST

type: **get**

URI Parameters:

userId - (number) - registered user id

RESPONSE

string

profile/status

REQUEST

type: **put**

required params:

status - (string), max length is 300 symbols - new user status (how many items will be returned in response)

RESPONSE

json object

[Default Operation Result Object](#)

profile/{userId}

get full user profile

REQUEST

type: **get**

URI Parameters:

userId - (number) - registered user id

RESPONSE

```
{
  "aboutMe": "я круто чувак",
  "contacts": {
    "skype": "skyp",
    "vk": "vk.com",
    "facebook": "facebook",
    "icq": "icq",
    "email": "email",
    "googlePlus": "gogep",
    "twitter": "twitter",
    "instagram": "instagra",
    "whatsApp": "watsap"
  },
  "lookingForAJob": true,
  "lookingForAJobDescription": 'Ищу работу, знаю это это и это',
  "fullName": "samurai dmitry",
  "userId": 2
}
```

profile

update logged in user profile. You should pass full object, if some properties are missing, then server will set default values (null or empty) for them.

So if you want only **aboutMe** property, you should send entire object: the same values for properties you don't want to change, and new value for property (aboutMe) you want to change

REQUEST

type: put

required params:

Full JSON object:

```
{
  "aboutMe": "я круто чувак 1001%",
  "contacts": {
    facebook: "facebook.com",
    github: "github.com",
    instagram: "instagra.com/sds",
    mainLink: null,
    twitter: "https://twitter.com/@sdf",
    vk: "vk.com/dimych",
    website: null,
    youtube: null
  },
  "lookingForAJob": true,
  "lookingForAJobDescription": 'не ищу',
  "fullName": "samurai d"
}
```

RESPONSE

json object

[Default Operation Result Object](#)

profile/photo

update profile photo

REQUEST

type: **put**

content type: **multipart/form-data**

how to upload photo with vanilla js:

```

3
4 <input type="file" id="photo"/>
5
6 <script>
7
8   function uploadPhoto()
9   {
10     var formData = new FormData();
11     var imagefile = document.querySelector('#photo');
12     formData.append("image", imagefile.files[0]);
13     axiosInstance.post('profile/photo', formData, {
14       headers: {
15         'Content-Type': 'multipart/form-data'
16       }
17     }).then(res => console.log(res.data));
18   }
19

```

RESPONSE

success response:

×	Headers	Preview	Response	Timing
▼	{data: {,...}, messages: [], resultCode: 0}			
	▼ data: {,...}			
	▼ photos: {small: "https://social-network.samuraijs.com/activecontent/images/user", large: "https://social-network.samuraijs.com/activecontent/images/users/15/us", small: "https://social-network.samuraijs.com/activecontent/images/users/15/us"}			
	messages: []			
	resultCode: 0			

FOLLOWING

Follow\unfollow users

follow/{userId}

REQUEST

type: **get**

URI Parameters:

userId - (number) - id of user which you want to check if you follow this user

RESPONSE

bool

follow/{userId}

Follow user by id

REQUEST

type: **post**

URI Parameters:

userId - (number) - id of user which you want to check if you follow this user

RESPONSE

[Operation Result Object](#)

follow/{userId}

Unfollow user by id

REQUEST

type: **delete**

URI Parameters:

userId - (number) - id of user which you want to check if you follow this user

RESPONSE

[Operation Result Object](#)

DIALOGS

dialogs/{userId}

start chatting, refresh your companion so that he was on top

REQUEST

type: **put**

URI Parameters:

userId - (number) - user id of your friend

RESPONSE

object

dialogs

get all dialogs

REQUEST

type: **get**

RESPONSE

object

dialogs/{userId}/messages

get list of messages with your friend

REQUEST

type: **get**

URI Parameters:

userId - (number) - user id of your friend

RESPONSE

object

dialogs/{userId}/messages

send message to your friend

REQUEST

type: **post**

URI Parameters:

userId - (number) - user id of your friend

required params:

body - (string) - your message to friend

RESPONSE

object

dialogs/messages/{messageId}/viewed

is your message viewed

REQUEST

type: **get**

URI Parameters:

messageId- (number) - user message ID

RESPONSE

object

dialogs/messages/{messageId}/spam

REQUEST

type: **post**

URI Parameters:

messageId- (number) - message ID to spam

RESPONSE

object

dialogs/messages/{messageId}

delete only for you, not for your companion

REQUEST

type: **delete**

URI Parameters:

messageId- (number) - message ID to delete

RESPONSE

object

dialogs/messages/{messageId}/restore

restore your message from deleted and spam

REQUEST

type: **put**

URI Parameters:

messageId- (number) - message ID to restore

RESPONSE

object

dialogs/{userId}/messages/new?newerThen={date}

return messages newest than date

REQUEST

type: **get**

URI Parameters:

userId- (number) - user id of your friend

date - (string) - desired date (string in date format)

RESPONSE

object

dialogs/messages/new/count

list of new messages

REQUEST

type: **get**

RESPONSE

object