FLYZOO CHAT - REST API V1.1

Reference guide – Release 2 (August, 29th 2016)

Contents

GET STARTED

API CONVENTIONS

- all requests are HTTPS POSTs
- fields (except arrays) are in string format.
- results are returned in json format
- a message {“status”:”error”,”result”:”error description”} is returned in case of error

EXAMPLE

Get the chat profile for the user with id = 1001.

curl -H "Content-Type: application/json" -X POST -d @getinfo.json https://api-v1.flyzoo.co/api/users/getinfo

getinfo.json is a simple text file containing the following data:

```
{"api_key":"fih2zp0wiznawcdqlqixs629acel8rci5pm0wjigjq67t2wcgs", "user_id":"1001"}
```

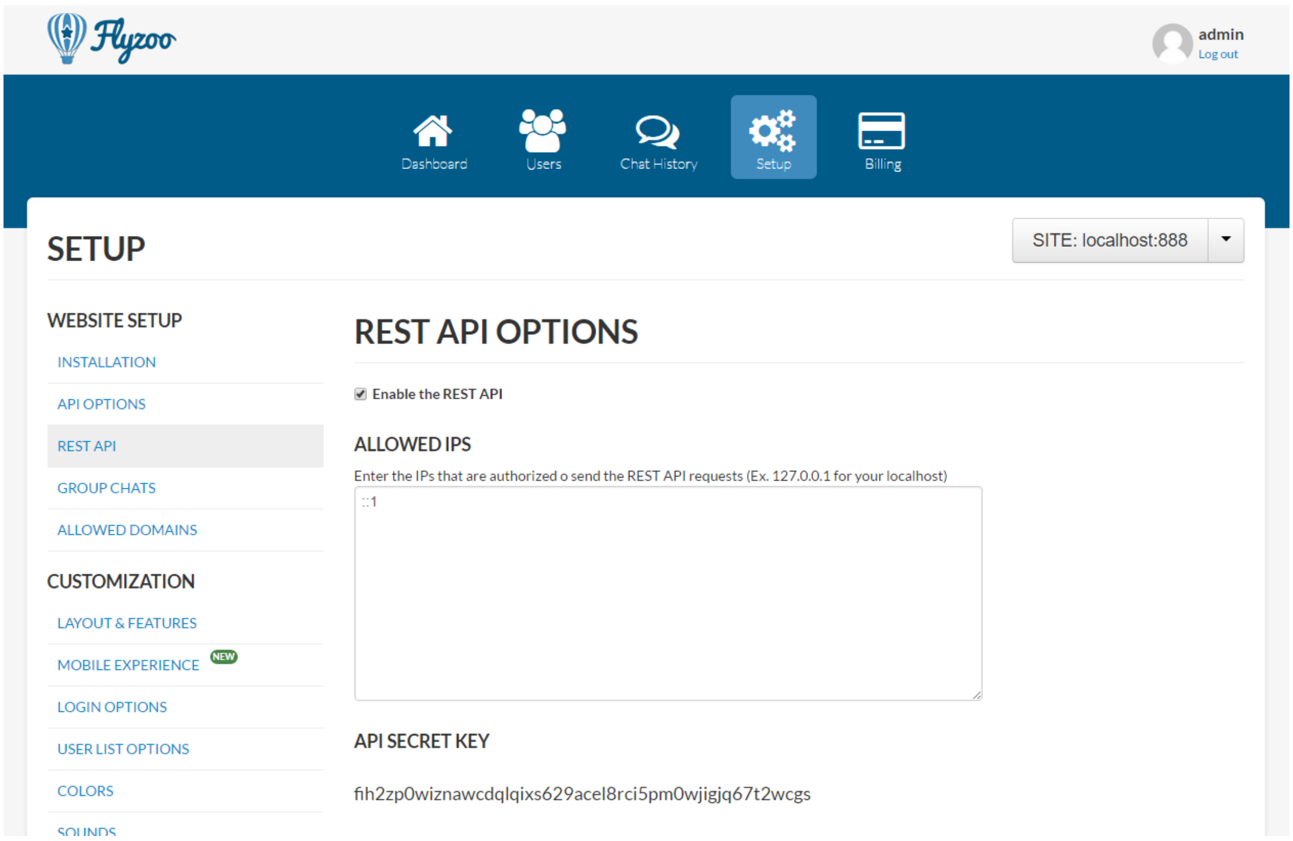
SAMPLE JSON RESULT

```
{
  "user_id":"1001",
  "user_roles":"operator",
  "user_profile_url":"",
  "user_avatar_url":"http://my-site.com/avatars/jake.jpg",
  "user_html_tag":"<strong>Magic Dog</strong>",
  "user_name":"Jake",
  "user_email":"jake@mydomain.local",
  "user_type":"2",
  "country_code":"",
  "country_name":"",
  "city_name":"",
  "status":"offline",
  "user_ip":"",
  "api_key":null
}
```

GET YOUR API SECRET KEY

First of all you need to retrieve your API SECRET KEY from the dashboard.
Each API call requires a valid **api_key.**

Log into the dashboard, go under SETUP > REST API and copy your API SECRET KEY.



SETUP

REST API OPTIONS

✓ Enable the REST API

ALLOWED IPS

Enter the IPs that are authorized o send the REST API requests (Ex. 127.0.0.1 for your localhost)

::1

API SECRET KEY

fih2zp0wiznawcdqlqixs629acel8rci5pm0wjigjq67t2wcgs

Make sure to whitelist the IP Addresses of the machines sending API requests.

You need to enter **::1** or **127.0.0.1** to send requests from your localhost dev environment, rejected requests will provide a message with the disallowed IP.

WARNING:

The API SECRET KEY must be exclusively used on server-side code, DON'T SHARE it and don't use it on client-side code.

PHP API EXAMPLES

The PHP API Examples have been designed to help you get started quickly with the REST and Javascript APIs.

The sample code will create a fake data set to let you test the most important features.

FAKE USERS (/users/create-fake-users.php)

User ID	Name	Type	Role
1000	Finn	User	basic
1001	Jake	Operator + Moderator	operator
1002	Marceline	User	pro

FAKE GROUP CHATS (/groupchats/create-group-chat.php)

Name	Allowed Roles	Allowed Users
Adventure Time	This is a public group chat	
Only Marceline		1002 (Marceline)
Only BASIC Role	Basic (only Finn has this role)	

Please note: While the admin, operator and moderators can access group chats even if they are not in the same Allowed Role, they won't be able to access group chats locked with the Allowed Users option if their User IDs are not listed there.

SETUP

To run the examples you first need to setup the **config.php** file:

```
$CONFIG_API_KEY = " --- your secret api key --- ";
$CONFIG_API_ENDPOINT = "https://api-v1.flyzoo.co/api";
$CONFIG_APPLICATION_ID = " --- your Flyzoo Application Id --- ";
$ROOT = "http://localhost:888/";
```

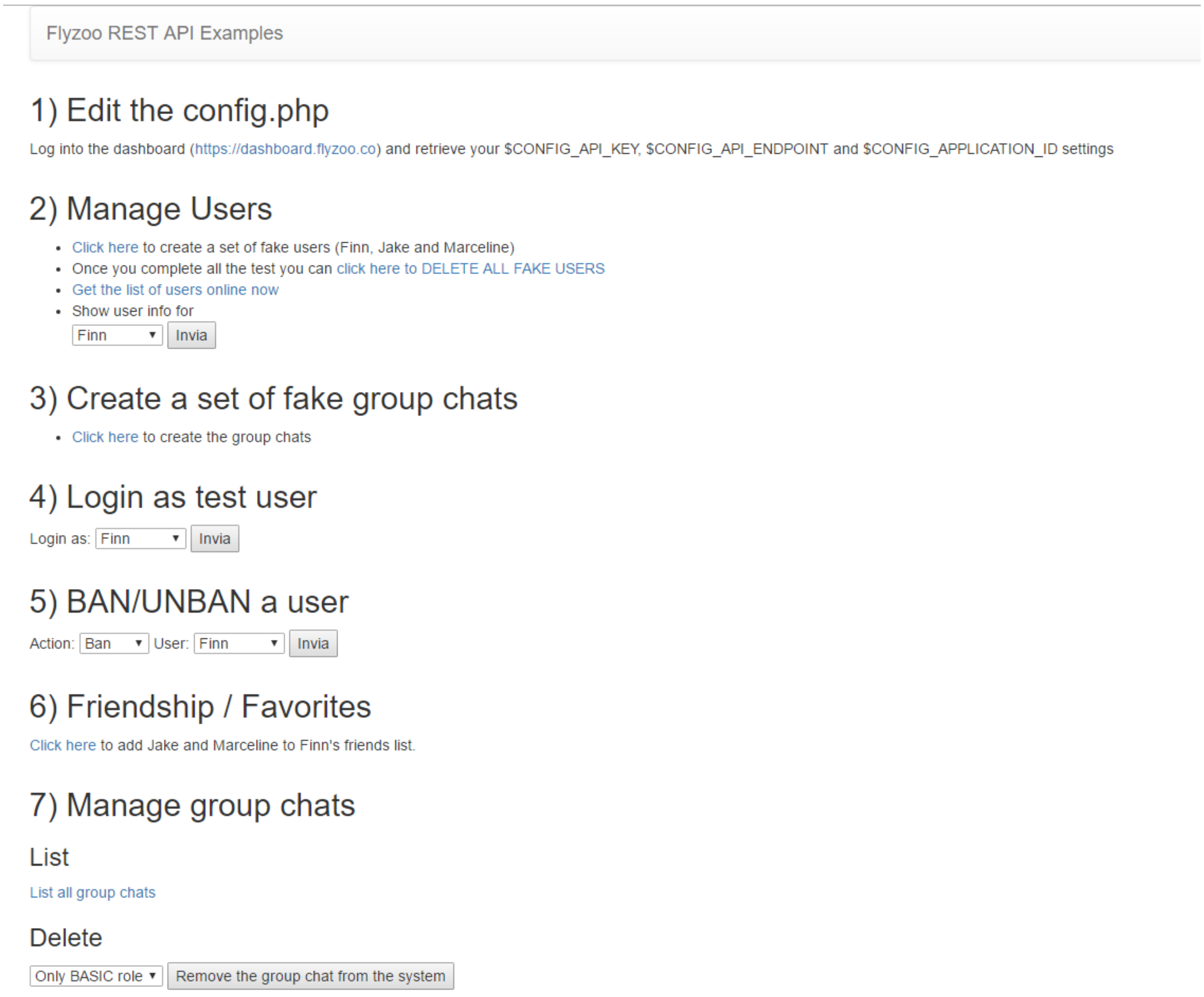
API-FRAMEWORK.PHP

The **api-framework.php** file provides a simple wrapper around the available API calls:
here you find some classes ready to use on your php projects - it's really simple to port this code to other languages like node.js, c#, etc.

RUN THE EXAMPLES

Let’s assume the sample project is running under http://localhost:888/.

Once you setup the **config.php** file and open the home page you should see the following screen:



Flyzoo REST API Examples

1) Edit the config.php

Log into the dashboard (https://dashboard.flyzoo.co) and retrieve your $CONFIG_API_KEY, $CONFIG_API_ENDPOINT and $CONFIG_APPLICATION_ID settings

2) Manage Users

- Click here to create a set of fake users (Finn, Jake and Marceline)
- Once you complete all the test you can click here to DELETE ALL FAKE USERS
- Get the list of users online now
- Show user info for

3) Create a set of fake group chats

- Click here to create the group chats

4) Login as test user

Login as:

5) BAN/UNBAN a user

Action: Ban User: Finn

6) Friendship / Favorites

Click here to add Jake and Marceline to Finn's friends list.

7) Manage group chats

List

List all group chats

Delete

Please make sure to run the following actions first:

- Create a set of fake users (Finn, Jake and Marceline)
- Create a set of fake group chats
- Login as test user (choose Finn to get started).

If everything worked as expected you can log into the Dashboard and notice that three users and three group chats have been created.

We suggest to use a two or three different browsers to emulate the users at the same time (you can also use Chrome and Chrome in incognito mode at the same time, for instance).

AVAILABLE API FUNCTIONS

The REST API endpoint is https://api-v1.flyzoo.co/api

Here is a summary of the available API functions; where not specified values are returned in the {“status”:”…”, “result”:”…”} JSON format.

USER MANAGEMENT

Example: https://api-v1.flyzoo.co/api /users/create

Function	**Path**	**Returned value**
Create User	/users/create	ok / error
Edit User	/users/edit	ok / error
Change Email	/users/changeemail	ok / error
Delete User	/users/delete	ok / error
Get User Status	/users/getstatus	online, away, invisible, hidden, busy
Generate Access Token	/users/generateaccesstoken	Access token value
Ban User	/users/ban	ok / error
Unban User	/users/unban	ok / error
Sync Friends	/users/syncfriends	ok / error
List online users	/users/online	Array of users (see user fields specification)
Get User Info	/users/getinfo	User (see user fields specification)

GROUP CHAT MANAGEMENT

Function	**Path**	**Returned Value**
Create a Chat Room	/groupchat/create	group chat id
List of All Chat Rooms	/groupchat/list	Array of group chats
Edit a Chat Room	/groupchat/edit	ok / error
Delete a Chat Room	/groupchat/delete	ok / error
List of all Users Online in a Room	/groupchat/online	Array of users (see user fields specification)
Publish Message in a Chat Room	/groupchat/sendmessage	ok / error
Delete History	/groupchat/deletehistory	ok / error

USERS

User types and roles are designed to manage the access level to features and group chats.

For example it's possible to disable the file sharing for guests, hide guests from the main user list or grant the access to a reserved chats only to members in the "paid" role.

User Types

User Type	Type Id	Notes
Guest	0	Guests can't be created via APIs, they are created automatically by the system when the chat widget is allowed to all users. (*1) Guests can change their temporary name, they are not allowed to upload a profile picture.
Users	1	Regular users don't have any special permissions
Admin	3	There is only one administrator – the user created to join Flyzoo. When online, the admin activates the live support system and is the only one authorized to customize the widget from the Dashboard.
Live Support Operator	5	Live Support Operators activate the live support system. When at least one operator is online and the widget is set to DOCK MODE for live support, the main button turns from "LEAVE A MESSAGE" to "WE ARE ONLINE". Operators can also access the Real-time User list from the menu icon on the dock. Operators can't moderate the group chats (ban users or delete messages)
Moderator	4	Moderators can ban users and remove messages but don't activate the live support system
Operator + Moderator	2	They activate the live support system and also can moderate the group chats – this is the highest level after the Admin

Notes

1) The widget can be completely disabled to guests by activating the "hide dock/widget to guests" option available from the Dashboard under SETUP > LAYOUT AND FEATURES menu

User Fields

Field	Description	Required
user_id	your local User ID, extracted from your database	X
user_roles	(ex. paid, pro, vip: single word, comma separated)	
user_profile_url	Url to the user profile	
user_avatar_url	Url of the user's avatar	
user_name	User Name	X
user_email	Must be unique as it's used as global primary key by the chat system. if you don't want to provide real emails you can use a format like username@my-domain.local	X

user_type	Guest ="0"
	User = "1"
	Operator = "5"
	Moderator = "4"
	OperatorModerator = "2"
		X

country_code	Ex. US = United States	
country_name	Ex. United States	
city_name	Ex. New York	
user_html_tag	This is a simple HTML label shown under the user name on the user list – feel free to user tags like <strong> or <img> to load small icons (14x14px)	
status	Readonly - Used by /users/getinfo method	
user_ip	required to ensure that the ban ip works correctly	

User Roles

With the user roles you can:

- Grant access to reserved group chats
- Show a group chat only to users in the same role
- Hide users not in the same role
- Allow only users in a specific roles to chat privately with other users and/or operators

Create

Just POST the request to https://api-v1.flyzoo.co/api /users/create

Edit / Delete

To edit a user just call the **/users/edit** method and pass the user data.
Make sure to pass all the data pulled from your database OR

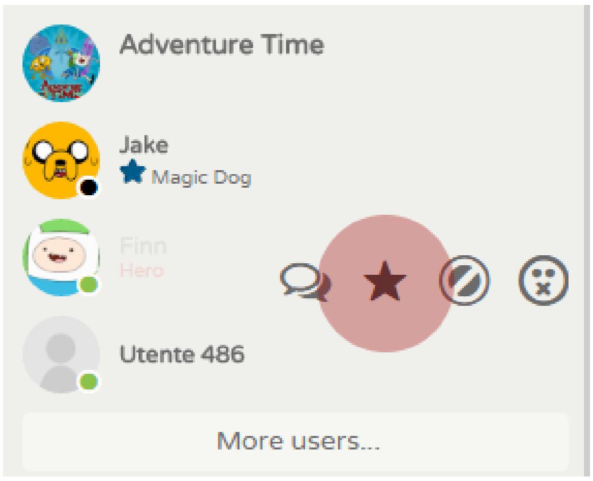
1) load the current users data first with the **/users/getinfo** method
2) modify the fields

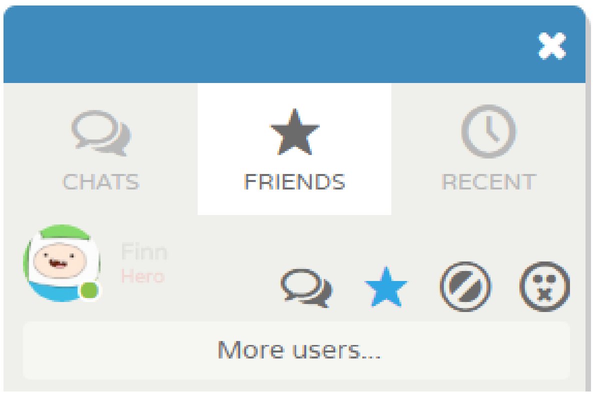
3) POST the updated data with the **/users/edit** method.

To delete a user just call the **/users/delete** with the **user_id** and **api_key** as usual.

Friendship

Flyzoo provides a Friends/Favorites list accessible from a tab on USER LIST.
Users can also befriend other users by clicking on the small "star icon" that appears hovering on a user name.





POST to the **/users/syncfriends** method to set friends for a user.

EXAMPLE:

```
{
"api_key":"fih2zp0wiznawcdqlqixs629acel8rci5pm0wjigjq67t2wcgs",
"user_id":"1001",
"friends":{"1002","1003"}
}
```

NOTES:

- The api will skip the entries of not found user ids
- Make sure to pass the entire list when removing/adding a single entry

User Status (single user)

POST to **/users/getstatus** to get the user status

EXAMPLE

```
{
"api_key":"fih2zp0wiznawcdqlqixs629acel8rci5pm0wjigjq67t2wcgs",
"user_id":"1001"

}
```

Return value:

```
{“status”:”offline”}
```

User Status (multiple users)

POST to **/users/getstatuslist** to get the user status for multiple users at the same time

EXAMPLE

```
{
"api_key":"fih2zp0wiznawcdqlqixs629acel8rci5pm0wjigjq67t2wcgs",
"user_ids": ["1000","1001","1002"]

}
```

Return value:

```
{"status":"ok","result":[{"user_id":"1000","status":"online"},{"user_id":"1001","status":"offline"},{
"user_id":"1002","status":"away"}]}
```

NOTES:

Please make sure to chat the responses whenever possibile to avoid excessive API requests, otherwise the server will throttle the number of available requests per second.

Login

There are two methods to login the users in to the chat

1) Use the Javascript API to pass all the user data
2) Use the REST API to retrieve an Access Token and pass it to the chat

This section covers the second method (the first one is described here: https://flyzoo.co/features/chat-api-integration)

STEP 1 – GET THE ACCESS TOKEN

First of all you need to generate an access token for the user with the **https://api-v1.flyzoo.co/api /users/ generateaccesstoken** method.

You just need to pass the **user_id** and **api_key** as usual:

{"api_key":"fih2zp0wiznawcdqlqixs629acel8rci5pm0wjigjq67t2wcgs", "user_id":"1001"}

You can call this function in your server code right after authenticating the user on your system.

STEP 2 – SAVE THE ACCESS TOKEN

Once you have the access token:

- pass it to the first page shown to the user after the login
- save it into the browser's localStorage object (field must be named "at")
- redirect the user to the main/home page of the site (add a small delay to make sure the localStorage object has saved the value).

SAMPLE CODE:

```
<script src="flyzoostorage.js" />

<script type="text/javascript">

var AccessToken = "<? echo generateAccessToken($_POST["user_id"]); ?>";

jQuery(document).ready(function() {
if(AccessToken == "") alert("invalid chat access token");

var m = window.FlyzooStorage; m.init();

if (m.storageMethod != "none") { m.storeValue("at", AccessToken); }

setTimeout(function() { window.location.href = "http://my-site.com" ,1000); }

});
</script>
```

Please check the **/users/login.php** example – flyzoostorage.js provides a simple library you can integrate with your code to manage the localStorage.

Flyzoo's main script will then detect the "at" automatically.

Ban / Unban Users

POST the user_id to **/users/ban** or **/users/unban** to ban/unban users.

The ban function will also block the user's IP Address – you can also unblock IPs from the dashboard (SETUP > IP BLACKLIST)

TIP: Make sure that you don't block yourself while running tests.

GROUP CHAT

Group chat fields

FIELD	DESCRIPTION	Required
group_chat_id	Group Chat Id – required to edit the group chat. This id can be retrieved with the /api/groupchat/list method.	
preview_max_width	Max Width (in pixels) of the embedded objects (like videos, images, etc).	
avatar	Url to an image used as avatar for the group chat	
name	Name of the group chat	X
manifesto	A short description of the group chat	
allow_guest	Allow guests to access the group chat	
embed_media_content	Set to true if the url must be converted to a media object (video, images, etc)	
enable_file_sharing	Enable the file sharing feature	
allowed_roles	Roles that can access the group chat. Ex. “paid, top, vip, moderators, team”	
guests_can_post	Set to true if guests can post, otherwise they can only read messages	
guests_must_change_name	Set to true if guests must change their username before entering the chat	
play_sound_on_join	Play a notification sound when somebody joins the group chat	
denied_access_message	Custom message for to users that can’t access the group chat (i.e. not in the same role, guests disallowed, etc). ex. "You can't access this group chat."	
enable_file_sharing_for_guests	Set to true if guests can upload files	
hide_user_list	Set to true to hide the user list for the group chat	
users_can_post	Set to true to allow users to post messages. Use this in combination with guests_can_post to create group chats where only operators and moderators can post messages.	
show_in_chat_list	Set to true if the group chat must be listed on the main chat & user list (this requires the widget to be in DOCK MODE or USER LIST MODE)	
user_list_background	Background of the userlist, if different from the default widget colors (ex. #ffffff)	

user_list_font_color	Font color for the userlist, if different from the default widget colors (ex. #000000)	
message_layout	3 Layout modes available for the messages: 1) FULL → show avatars and round message bubbles 2) NO avatars → Reduced space 3) COMPACT → No avatars, no round bubbles – text on full row	
user_list_start_minimized	Set to true if the user list must be minimized by default; users can toggle the user list at any time	
hide_to_other_roles	Set to true hides the chat from the user list to users not in the allowed roles.	
user_message_background	Background color for the current user	
user_message_text_color	Font color for the current user	
other_user_message_background	Background color for messages sent by other users	
other_user_message_text_color	Font color for the messages sent by other users	
order	Appearance order on the main USER & CHAT LIST	
allowed_users = array("");	List of User Ids allowed to the access the group chat	
hide_to_other_users	Hide the chat from the user list if the current user is not in the allowed users list	
enable_history	Set to true to show the "Load more messages…" button	
url_filter_mode_guests url_filter_mode_users	Set what users/guests can post "0" → can post all links "1" → can post only links from the specified domains "2 →can post all links except those listed "3"→users can't post any links	
url_filter_list_guests url_filter_list_users	Use this in combination with url_filter_mode_guests and url_filter_mode_users. Ex. If guests can only post content from Youtube set url_filter_mode_guests to "0" And set url_filter_list_guests to "youtube.com"	
top_banner_html	HTML/Javacript code to render a custom banner and run custom code at the top of the group chat	
show_offline_users	Set to **true** to show the users currently offline	

Create / Edit / Delete Group Chats

Please check the **api-framework.php** to see the default values: "name" is the only mandatory field to create a group chat.

Once you post to **/groupchat/create** the system will return a **group_chat_id** – you can store this value to delete/edit the group chat in the future or post messages.

Editing and deleting a group chat follows follow the same steps described for the users.

Post a Message

The **/groupchat/sendmessage** allows you to post a message to a group chat.

The message is saved and sent in real-time to all users that joined the group chat.

EXAMPLE

```
{

"api_key":"fih2zp0wiznawcdqlqixs629acel8rci5pm0wjigjq67t2wcgs",

"group_chat_id":"52b18bfa53e51f040cd0b19c",

"user_id ":"1001"}

"message ":" <strong>Hello!</strong> "

}
```

JAVASCRIPT API

CHECK THE USER STATUS (FlyzooApi.getUserStatus)

SYNTAX: FlyzooApi.getUserStatus(<LocalUserId>,<SuccessCallback()>,<FailCallback()>)

EXAMPLE

```
var UserID = “1001”;
var UserName = “Jake”;

FlyzooApi.getUserStatus(UserID,

            function(e) {

               alert(UserName + " is "+ e)

            },

            function(e) {   alert("error: " + e)   })
);
```

OPEN A GROUP CHAT

To launch a group chat you need to know the Group Chat Id. The API will provide you this value when you create the chat – it’s a good idea to store it somewhere on your database, otherwise you can retrieve it by listing the available group chats with the https://api-v1.flyzoo.co/api /groupchat/list method.

SYNTAX: FlyzooApi.startGroupChat(<GroupChatId>,<Label>);

EXAMPLE

```
var groupChatId = “573c4135a59c353ef0069b27”;
var groupChatLabel = “The amazing group chat”;

FlyzooApi.startGroupChat(groupChatId, groupChatLabel);
```

START A PRIVATE CHAT WITH A USER (FlyzooApi.startChat)

SYNTAX: FlyzooApi.startChat(<LocalUserId>,<SuccessCallback()>,<FailCallback()>)

Success and fail callbacks are helpful for various scenarios.

For example you might want to show a spinning loader inside a "CHAT WITH JAKE" button and then hide it when the API starts the chat.

Note: pass function() { } if you don't wont to perform any action on callback.

Possible Fail messages:

not-found → user not found
cant-chat-with-yourself → you are sending your same current id
local-userid-not-provided → id set to blank
access-token-missing → authorization error
system-error → connection issues and other api errors.

EXAMPLE

```
<a id="btnStartChat" href="#" onclick="javascript:FlyzooApi.startChat(1002, startSuccess,
startFail)">start chat with other user </a>

<script type="text/javascript">
   function startSuccess(e) {   alert("success");   };
   function startFail(e) {   alert("fail with message:" + e);  };
</script>
```

USER LOGOUT

To log out the user from the chat you just need to remove the following localStorage entries:

at
chat-status
fz-api-hash
fz-friends-hash
lastcmd
hdtg
cid

The FlyzooApi.resetSession() does that for you.

For example on could run this code on the button that performs the logout on the site

```
$("#logoutButton").click(function() {

    // custom logout logic here
     FlyzooApi.resetSession();
     window.location.href = "url to exit page here….";


}
```