AtMe Documentation

INDEX

INTRODUCTION

AtMe is intended to be a simple software to create Discord bots (1) with graphical programming (2). AtMe aims to reduce the level of skill required to make Discord bots. Grasping external functions, AtMe generates inerrant javascript code. It is important to note that AtMe does not depend on Discord and can run separately. Please view the one minute *example video (EXAMPLE VIDEO)* to gather a simple visual and example of AtMe functions *(Note: All functions aren't shown in the video to reduce lengthy content).*

GETTING STARTED

On launch, the user is granted with *image A*. The plus button *(+)* creates a new workspace, the path of the workspace defaults to \%USER_HOME%\AtMeWorkspaces\.

CREATION:

There are five values that should be set before creating the bot.

- Name - This defines the name of the *bot (4)*, may be referred to as the id. This value is often condensed and spaces may be removed. *(Note: This value is unchangeable)*
- Description - This defines the description of the bot and is only used in the manifest.

- Token - This defines a value that is severely important and is used to connect AtMe/Discord-bot to Discord. A token can be granted from Discord Developer Portal. *(Note: This value is not shared and is only saved remotely in "index.js")*
- Prefix - This defines a constant value.
- Author - This defines the author of bot and is only used in the manifest.

To finalize the bot the user should click on the AtMe icon. It is important to note that all functions of the bot will work even if these values are "invalid" (except the run *[RUN BOT]* function).

FIRST BUILD:

Upon creation AtMe should recognize that there is no client wrapper, note that this is normal on first build. Upon pressing "Ok" in the popup, AtMe will generate a new wrapper *(Note: There is a rebuild option in the File menu bar for future rebuilds)*. During the rebuild a command prompt will open and upon completion the user should close it. Afterwards, AtMe will finish arranging the files and the user should be prompted with completion *(Note: It is normal for this process to take up to 5 minutes, and until this process is completed the app will be responsive but no functions will work)*. After completion the bot should run and all functions should work.

INSIDE WORKSPACE

AtMe has three options: refactor bot *(REFACTOR)*, add elements *(ADD ELEMENT)*, run *(RUN BOT)*. These options allow the user to manipulate and customize their bot.

REFACTOR

At any moment the user can refactor and change maniputable values. To refactor the bot the user should be viewing the *workspace (4)* screen, viewable in *image B*. To open the refactor bot dialog the user should press the settings icon/button.

ADD ELEMENT

The user can add elements to his bot as long as the bot is properly built. To add elements the user should be viewing the *workspace (4)* screen, viewable in *image B*. To open the element dialog the user should press the plus icon/button. Upon opening the element dialog the user should be granted with all elements available in AtMe. The default element types in AtMe are:

Event:

The event bit is the main bit of AtMe and used to handle Discord events. Utilizing *Blockly's (3)* features, the user can create events to manipulate guilds, channels, messages, and users.

Template:

The template bit is used to create templates for advanced messages. In the template editor, the user is granted 5 paint tools to fit their needs. The static elements in the left column are variables, and are manipulated to their respective values in runtime. Alone, the template will have no effect. You can utilize templates in an event bit, specifically the "Send template" block under the "Message" blockly category.

After saving an element (saved upon closing the windows), the respective files are saved in the “/bits/” directory of the workspace.

RUN BOT

At any moment the user can run his bot. To run the bot the user should be viewing the *workspace (4)* screen, viewable in *image B*. Next, the user should click the play icon/button. This should open a command prompt (console) and a white button. During this time the bot

will be running and will function. To test and use your bot, you'll need to implement the bot in Discord and use Discord's respective properties *(Note: This is optional due to AtMe not depending on Discord, but is required to view the bot's elements in action, hence, the user may need prerequisites of Discord knowledge which can be learnt online at Discord's website).* The user can stop their bot from running by closing the command prompt and pressing the white stop button. This will return the user to the workspace screen.

TERMS

(1) Discord bot - Discord offers an API for automated users. See more at: [Discord Developer Portal — Documentation — OAuth2]

(2) Graphical programming - Graphical programming includes but is not limited to Blockly. Graphical programming consists of simple and meaningful code generation through visual means.

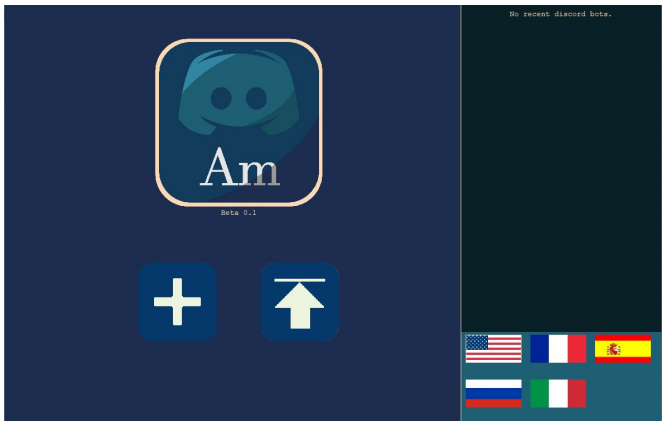
(3) Blockly - External library. See more at: [Introduction to Blockly | Google Developers]

(4) Bot | Workspace - workspace of graphical discord bot.

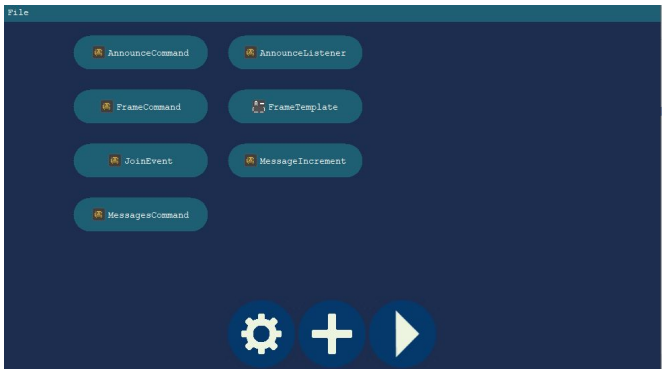
IMAGES

(Note: Images may not be exact and updated, however if the app looks abnormal, contact us)

A:



B:



EXAMPLE VIDEO

An example of AtMe being used can be found at: https://www.patetlex.com/res/vid/atme.mp4

(Note: The token displayed in the video has been changed)

Contact: support@patetlex.com

Written by: Tye Phoenix

Date: 24 February 2021

Updated to AtMe version: 1.1.0.0

In regards: AtMe Documentation

From: PatetLex#19117