Olivia Kisker, Sam Turecamo, & Samantha Greenberg
Professor Fudge
IST 256 - Python Project
May 4, 2017

Emoji Vote

Our Python Code Explained

Emoji Vote generates a voting system that can be used for fundraising purposes. Users of this program will need 3 types of accounts in place: Venmo, Gmail, and Plot.ly. Venmo is a popular banking app with the MIllennial generation because it is linked directly to their online banking account and allows friends to send payments to each other. Gmail is for email and in this case is used in association with the Venmo account to make the program run. The Gmail must match the Venmo account email or the code will not work properly for the user. Plot.ly is used to create a visual bar graph of user participation when they send in cash to vote. The program will prompt the user for information about all three accounts. After responding to the gmail and plotly login, the plot.ly graph should appear and reflect the cash amounts for each Emoji vote (looks best in full-size).

Venmo sends an email when a payment is received. Our code looks through email to find the exact money amount and the choice of emoji. It then it adds the money and emoji to the connected plot.ly graph. The height of the bars reveals how much money has gone into each emoji.

Our code starts by importing modules such as email and plotly. We start off by making two lists- one for the money amount and one for the emoji. The gmail API asks for the username and opens up the “Inbox” folder.

We have two functions- the open files function and the process mailbox function. The open files opens files, then splits the words.

Process mailbox searches for emails and separates the subject and message. The code splits the string in both. Our first step is to find the money or dollar amount contributed. We do this by finding matching words in the subject and message and picking out the word that starts with ‘$’. It adds this to the money list. To find the emoji, they are formatted in the email as utf-8 with html around them, like for example: =F0=9F=98=81 . The code looks in the message for a word that starts with ‘ =’. We had to translate the emoji from this utf-8 format to a normal emoji. To translate the emoji, we have to open the file utf8_emoji.txt and lookup the utf-8,

replacing it with the emoji. It adds the emoji to the emoji list. The two lists are combined into a dictionary, adding the amount of money if the emoji is the same. For plotly they must be formatted in list, so the dictionary is converted back into two lists.

After the process mailbox function, we ask the user the password to their gmail. If the email output is ok, we call the process mailbox function.

To make the bar graph, we plot the emoji on x and the amount of money on y. The title, size, and color of font is edited. It is saved as emoji_new. We close the gmail inbox and the plotly graph appears. If the email output is not ok, like if you type the wrong password, it prints an error about not being able to open the mailbox. We logout of gmail.

Team Member Contributions

Olivia Kisker

- Came up with the innovative idea for the project
- Wrote bulk of the code
- Added helpful code-breakdown notes within the code
- Helped edit code explanation

Sam Turecamo

- Conducted research for the program's APIs
- Tested code to make sure it worked
- Arranged and formatted the poster layout
- Printed and picked up the poster

Samantha Greenberg

- Researched Emoji UTF-8 and edited the program's Emoji file
- Gathered poster materials
- Contributed to the poster design and composition
- Wrote code explanation