DRIFTER : TWITTER GAME REPORT

Written by : Zachary McGrew, HildigerR Vergaray, Ray Weiming Luo, Sereyvathanak Khorn

Last modified on : May 31, 2016

IMPLEMENTATION

Our project was to implement a science-fiction adventure game inspired by the concepts of the popular gaming stream Twitch Plays Pokemon[1] using the Twitter API. User's would be able to send in commands via Twitter by sending in tweets to a bot Twitter account.[2] The bot will wait five minutes before it will tally up all the tweets and display the top five most tweeted commands to the bot. The highest voted command will be executed in the game and the bot will post a picture of the status in the game. The bot has a flexible way to handle commands, the user can send either, '3', 'three', or 'third' which will all be translated to mean the value three. Also, the bot will reject any invalid commands and will only execute commands that are possible to perform in the game in the specified scenario the player's are in. An example is that players can not harvest from a planet if they have not orbited a planet first. The list of possible commands in the game are : orbit <planet number>, depart, drift, home, harvest, jettison [#] <item>, buy [#] <item>, sell [#] <item>, attack, refine [#] <item>, gamble [#], repair, craft [#] <item>. The game will also list the available commands for the users to perform which gives them a starting point of the available moves.

The project includes a website[3] which can give players various information about the game such as a list of all the commands along with a brief description of what the commands will do. The website serves as a resource material for users to obtain more information about the game in an organized and precise place. It includes the photo of the latest post showing the status of the game, a brief story of the game and a scoreboard of the top active players with the username and their corresponding total number of tweets, successful tweets, days played, and the date of the last time played. Also, There includes a list of craftable items that can be made in the game along with a list of the required resources the player's must have in their cargo in order to successfully craft the item.

USER EVALUATION/EXPERIMENT METHODOLOGY

From our experiment, the users tested the game using their own Twitter and in their own environment to mimic the comfortability of using the system in a natural way. The experiment consisted of four players who played the game on their own pace for one day and the next day they haven't played the game. This is to determine the correlation between the Twitter usage of the players when they are motivated to play a Twitter based game and only interacting with Twitter by posting their status. Our interaction with the players during the experiment was online communication. This version of communication with the users is what we would ideally expect if more players were to play the game and the community with interact mostly online to determine strategies, plans, or any ideas about the game in general.

The experiment consists of users sending tweets to the bot to perform in-game, any invalid commands would simply be ignored as a basic error check. And the game will operate on a tally system of the most highly tweeted command will be performed. We wanted the user to explore the game on their own and would only tell them the basics of how to tweet to the bot.

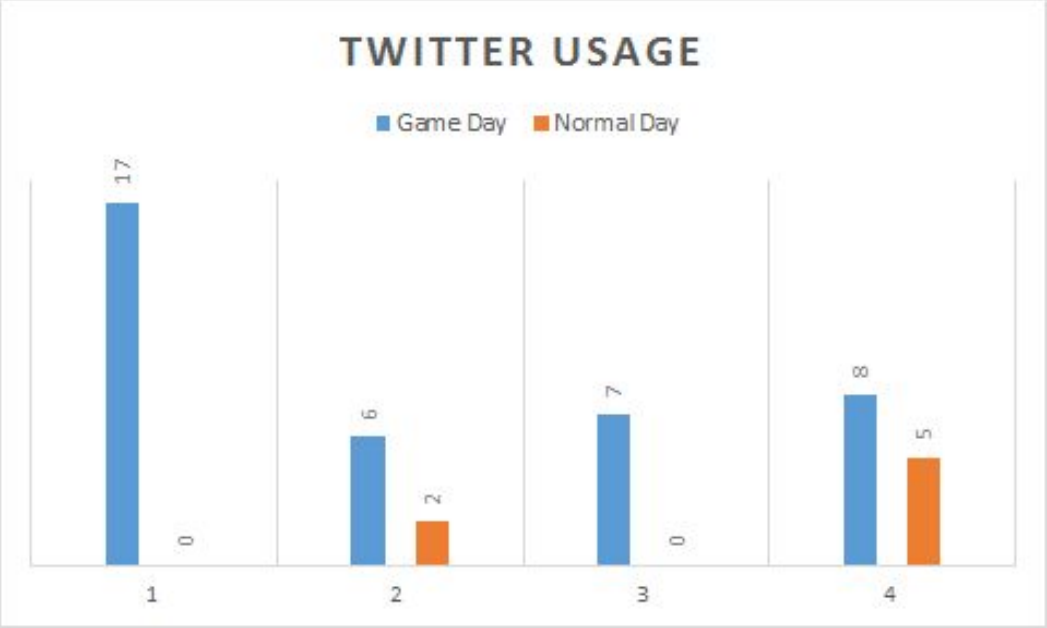
---

[1] Link to Twitch Plays Pokemon webpage : https://www.twitch.tv/twitchplayspokemon
[2] Link to Drifter Game Twitter account : https://twitter.com/DrifterGame
[3] Link to the Drifter Game website : http://sw.cs.wwu.edu/~mcgrewz/

However the benefits of the website is there to help guide the new players and give them an overview of the possibilities that they can do in the game. What we observed is that the users will only send in tweets of the commands that are listed under *Available commands*. Since the users know that these are the commands that they are restricted to, it would be wasted effort to try and send a command which will not count as a vote in the game. Another observation we've noticed is that the time period may be slightly long and the users will have a lot of down time before they can send in another valid tweet to the bot. Initially, we thought five minutes was a good time for users to read the current status of the game, think of a command they want to perform in-game, review any previous changes in the game from where they last played, and change their vote. However, sometimes the five minutes may seem extensive if a user is confident with their vot. The user would have to wait a long time before they can send in another tweet to the bot without having their previous tweet overwritten.

SUMMARY OF DATA

After conducting the experiment, we have seen a large difference of the user's Twitter usage on the day of playing the game and the following day when they didn't play the game. Below is the chart of Twitter usage from the four participants of the experiment. The experiment was conducted on the weekend, Saturday (Blue) being the day of playing the game and Sunday (Orange) being the day that user's didn't play the game.



As we can see from our data, there is a greater Twitter usage from the users who had the game to play on Twitter in comparison to their normal Twitter usage the next day. User 1 sent out 17 tweets and User 3 send out 7 tweets which were all directed to the bot. User 1 and 3 shows signs of only using Twitter to play the game and would not send out tweets the following day if the users weren't playing the game. While, User 2 sent out 6 tweets and User 4 sent out 8 tweets to the bot on the day of playing the game. And the next day, User 1 sent out 2 tweets and User 4 sent out 5 tweets which were not directed towards to bot. This still proves a greater amount of tweets being used towards playing the game than their average Twitter usage.

Therefore, we can conclude that our game gives the Twitter users more reason to interact with Twitter and tweet out to the bot than simply only posting their normal statuses.

DISCUSSION

After the experiment, we overcame various obstacles due to the limitations when using the Twitter API[4]. Some of the limitations include the Twitter policy of the amount of tweets an account can send per minute. We calculated that Twitter limit each user to 2400 tweets a day which is equivalent to 1.6667 tweets per minute. To be safe with our Twitter bot, we set a cap of five minute segments between each tweet our bot would send out to follow Twitter's guidelines and avoid being banned from their servers. Also, there is a hidden limitation for developers using the Twitter API when retrieving the tweets that were sent to the bot. We were only able to get fifteen requests of the tweets sent to the bot every fifteen minutes. This limits the bot to retrieve the data to every other minute. This limitation led to the idea of having a democracy voting system as it will set a time period for users to tweet to the bot and tally up all the votes to determine which is the most popular command to execute in the game. Additionally, the democracy system will include an accurate system to determine which command should be executed in the game by removing any outlier tweets. Thus the majority of the players want the perform the same action and the majority will agree on the vote.

One of the challenges we overcame for the game was determining the validity of the tweets. Users can send a command to the bot in various way and can be logically correct but syntactically different. An example can be shown as, User_1 sends a tweet to the bot with the message, "I would like to orbit 3." And User_2 sends a tweet with the following message, "Orbit the second planet." Both are logically equivalent but the syntax is different, the program will determine that both are the same command and will handle both commands the same way using regular expressions. Another challenge we faced was replacing the user's tweets to their most up to date tweet sent to the bot. The user can send any number of tweets at any time period. However if they send multiple tweets between the five minute waiting period the bot will take their most recent tweet to avoid a spam of tweets from the same user. An example is User_1 can send "Harvest on this planet", multiple times to skew the voting to User_1's favor. To avoid any skewed data and provide a fair and accurate democracy system, the bot will determine that the user has sent a previous tweet and will replace his/her tweet to the most recent one.

Some improvements that can be made in the game is to make the game more in-depth and provide more functionalities in the game. Although we completed a finished product that performs the actions we expected when designing the game. With additional time, we can implement a more in-depth game and have more functionalities that users can perform in-game. Additionally, with more time we can include customized graphical designs for the items in the game to make it more visually appealing.

[4] Link to the Twitter Rate Limits : https://dev.twitter.com/rest/public/rate-limits