Tom Jennings
29 May 2012
tomj@wps.com
323-620-3801

Twitter Punisher

Software application development specification

This is an informal proposal for software construction based upon my understanding of the task. I'll just state this all as fact for expedience, but I actively solicit your corrections and suggestions.

The overall goal is to accept Twitter messages from the internet, process them as discussed and have the machine Brett is building bang typewriter keys onto the display of a smartphone. I am responsible for the software that accomplishes this, from internet yanking of tweets to solenoid motion within the machine.

I will be provided with a windows computer, some connection to the internet (wifi?), and USB thumb drive(s) for routine extraction of the log files.

The software as I see it must accomplish the following:

OVERALL

- Design goal: turn-key operation, hopefully as basic as 'power strip' on/off. As much as time and budget allows the system will be made immune to USB cable disconnect, power failures, etc without human intervention.

TWEETS

- Fetch tweets from specified account via a pre-existing internet connection and Twitter account (login and password).

- log (csv spreadsheet) every twitter. These are numbered sequentially. logging is largely TBD; I would suggest one log file per day or other solution.

- Log written to local disk, and also to USB stick. USB stick can be removed or reinserted at any time.

- Any other processing to log file(s) post-facto the responsibility of the client, is done offline and not within this program.

- Tweet serial number accuracy maintained across program restarts/reboots and across log file

rollovers/deletions.

- unique, additional action for every 100th tweet received. This action is TBD as of 28 May and could require additional work not covered by this agreement.

BEHAVIOR (User experience)

- Presentation and "pile up" handling controls -- how to handle incoming tweets to meet subjective audience experience requirements which are largely TBD. eg. the typewriter can handle 1 - 2 characters per second but may arrive at 10, 100, or more per minute. Attention span limitations vs. typing accuracy. Initially at least this will be "scenario b" as per email; entire tweets will be typed (possible max character limit) but during pileups only the first N characters of each tweet will be printed. The algorithms here are still TBD and will require tuning.

- Flexibility in code to accomodate other possible scenarios. Scenario changes will require coding design, development and debug cycles. Those changes should be anticipated as much as possible in initial design.

- Bell rung at the start of every tweet.

HARDWARE

- Nixie counters accurately reflect currently printing twitter number. This is the same as the persistent serial number.

- Characters within each tweet will be mapped to an appropriate typewriter key where possible. Text will be stripped of extra white space. Unicode characters will be arbitrarily mapped to typewriter keys. Non-text content will be silently dropped (images, sound, etc). Case is ignored. ASCII graphics and digits will be mapped as the hardware allows.

- Typing of characters will be as fast as possible within the limitations of the hardware; at this time this seems to be 1 or 2 characters per second.

- Short pauses in typing will be imposed as necessary to create the user experience.