DESIGN BY TWEET: PROCESSING-INSPIRED COLLABORATIVE CREATIVE CODING VIA TWITTER

BRAD TOBER

Boston University, Boston, Massachusetts, USA

brad@bradtober.com

Keywords
Programming
Creative code
Twitter
Social media
Collaboration
Design
Processing

Design by Tweet enables Twitter users to collaboratively code visual compositions by writing tweets that use a simple Processing-inspired API. Twitter users can participate by sending a specially formulated tweet to @designbytweet. In response, this account will automatically post an image of the composition after executing the user's commands. The *Design by Tweet* name mirrors that of *Design by Numbers*, one of John Maeda's projects at the MIT Media Lab in the 1990s.

2016.
xCoAx
.org

Computation
Communication
Aesthetics
& X
Bergamo, Italy

1 OVERVIEW

This paper introduces *Design by Tweet* (DBT), a tool enabling collaborative, Processing-inspired creative coding via Twitter (video demo). DBT is, at its essence, a Twitter bot that continuously listens for @reply tweets to itself (tweets that begin with "@designbytweet") via the Twitter API. Hosted on a Node.js server, DBT parses each of these @reply tweets and first makes a determination as to whether or not it is in the required format. If the tweet is able to be parsed according to the DBT format, its commands are then executed and the resulting visual output is applied to a persistent server-side canvas / drawing surface. DBT then responds by posting a tweet with an image of the canvas following the execution of the user's commands, mentioning the user in the tweet text.

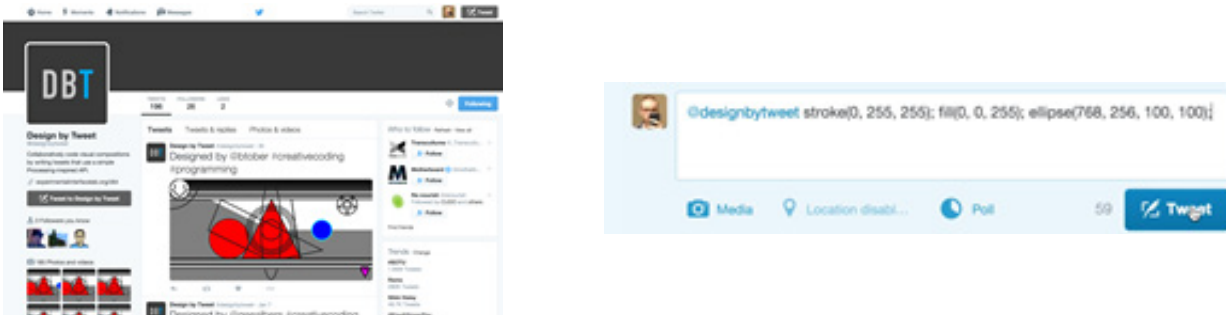

Fig. 1. Still shots of the footage taken in Snowdonia.

While many projects (like Processing) have made creative coding significantly more accessible, the need for standalone integrated development environments still presents a barrier to entry. DBT aims to address this issue by appropriating an existing platform that is conventionally used for social communication and interaction. Twitter's prominence means that using it as a platform for creative coding makes logical sense in terms of increasing the potential number of creative coders.

2 TECHNICAL DOCUMENTATION

The collaborative nature of DBT means that multiple users simultaneously contribute to a single canvas. The DBT canvas is a fixed size, with a width of 1024 pixels and a height of 512 pixels (for optimal visibility on Twitter). The origin, where the x- and y-coordinates are both equal to 0, is located at the top-left corner. From the origin, the x (horizontal) axis increases toward the

right and the y (vertical) axis increases toward the bottom. DBT users can view the current DBT canvas, as well as a historical record of all previous canvas iterations, on the @designbytweet Twitter profile.

Users can participate by sending a specially formulated tweet to @designbytweet. The format of this tweet should follow "@designbytweet command1; command2; command3; ..." where "@designbytweet " is the DBT Twitter handle, followed by a space, and "command1; command2; command3; ..." is a sequence of one or more DBT API commands, each followed by a semicolon. Spaces are not needed between both commands and the comma-separated parameters of commands. Users may include as many commands in a tweet that can fit within Twitter's 140-character limit.

The DBT API is "Processing-inspired" in the sense that it reflects the programming syntax that the Processing project has developed and promoted (Processing). However, the functionality of the DBT API is limited to a subset of the Processing API. Many functions, as well as code elements like variables and loops, are not currently implemented. Also, all parameters must be literal values. The initial DBT API incorporates some of the most useful commands for quickly generating visual output (full documentation and API reference available at http://www.experimentalinterfacelab.org/dbt).

3 CONTEXT

While legitimate, interactive Twitter bots respond to some sort of structure within a human Twitter user's tweet — for example, @KLMfares (Twitter 2015) — one would hesitate to refer to this as programming the bot due to the fact that the interaction does not occur over a series of back-and forth-tweets (a "conversation" in Twitter terminology). This is even true for the only other (to the author's knowledge as of this writing) example of programming via Twitter, Wolfram Research's *Tweet-a-Program* (Wolfram 2014). This Twitter bot allows users to tweet Wolfram Language programs to @wolframtap and receive the output in reply. However, these programs must exist within a single standalone tweet, and there is no collaborative element integrated into the system. As an example of a broader programming-related use of Twitter, users of the SuperCollider language for real-time audio synthesis and algorithmic composition share short, self-contained programs via the #sctweet and #supercollider hash tags (although these programs are not executed on Twitter itself).

4 FURTHER WORK

DBT currently exists as a provocation — a number of Twitter users have demonstrated its technical viability, but its conceptual viability remains to be seen and can only be tested with the broader audience that would accompany increased visibility of the project. The author hypothesizes that this visibility may come along, in part, with increased utility of DBT, and so expanding the DBT API is a priority for further work on the project. Several early DBT users expressed particular interest in the ability to repeat code easily, such as with a "for" loop. Moving forward, it will be necessary to identify those elements of the Processing API that can successfully be translated to and used within the context of Twitter. For example, certain aspects of the Processing API that use commands to encapsulate other related statements — like beginShape() and endShape(), or pushMatrix() and popMatrix() — may not be appropriate for use by DBT given its collaborative focus. A user using these commands in a way similar to Processing might find his or her efforts interrupted by another user attempting to contribute to DBT at the same time. Accordingly, all future improvements and additions to DBT must be approached with the objective of facilitating user collaboration in mind.

REFERENCES

Processing. "Language Reference (API)." Accessed 13 Jan 2016. http://processing.org/reference.

T**witter.** "Automation rules and best practices." Last modified 8 Apr 2015. http://support.twitter.com/articles/76915.

Wolfram. "Introducing Tweet-a-Program." Last modified 18 Sep 2014. http://blog.wolfram.com/2014/09/18/introducing-tweet-a-program/