Open source communities for artists
Lessons learned from the Tracery Project

Dr. Kate Compton
CROSS Incubator Fellow

Tracery: generate text, graphics and more

Tracery is a super-simple tool and language to generate text, by GalaxyKate. It’s been used by middle school students, humanities professors, indie game developers, professional bot makers, and lots of regular folks.
In open source, we feel strongly that to really do something well, you have to get a lot of people involved. -Linus Torvalds
Part I

About “contributors”
Who is an open source contributor?

- People who write the code
- People who write the documentation
- People who port the tool to other languages
- People who make connecting tools, pipelines, platforms
- People who write tutorials or teach workshops
- People who fundraise, organize or provide long-term stability
- People who make art with the tool
Who is an open source contributor?

- People who write the code
- People who write the documentation
Who is an open source contributor?

- People who port the tool to other languages
Who is an open source contributor?

- People who make connecting tools, pipelines, platforms

@v21 George Buckenham
Who is an open source contributor?

- People who write tutorials or teach workshops

Shawn Graham (@electricarchaeo)

Aaron Reed (@aaronareed)
Who is an open source contributor?

- People who fundraise, organize or provide long-term stability
Who is an open source contributor?

- People who make art with the tool

@jonnyIun
@jonnyIun

hi everyone, i made a quick bot for myself with tiny actionable self-care reminders bc i've been obsessively on twitter lately: @tinycarebot

replies

Followers

Likes

48m

@DanHett
@DanHett

Tracery is honestly my favourite thing ever. i make it write music and play in nightclubs for me! :D

Dan Hett
@DanHett

Tracery is honestly my favourite thing ever. I make it write music and play in nightclubs for me! :D

replies

Followers

Likes

48m

@GalaxyKate
@GalaxyKate

So thanks again for CBDQ; @InstrumentBot has been like a stout-hearted little robot friend who reminds me to make art I like.

replies

Followers

Likes

48m
Who is an open source contributor?

- People who write the code
- People who write the documentation
- People who port the tool to other languages
- People who make connecting tools, pipelines, platforms
- People who write tutorials or teach workshops
- People who fundraise, organize or provide long-term stability
- People who make art with the tool

Not on the GitHub repo!
Convening for open source tools for the arts

(June 2018, Minneapolis)

- processing.org
- three.js
- wekinator
- openframeworks
- tracery
Kinds of outreach

- How can you reach people you don’t have access to?
  - Different departments
  - Different fields
  - Different language
  - Different needs
  - Age/interest/community
Kinds of outreach

An Introduction to Twitterbots with Tracery
Shawn Graham
An introduction to Twitterbots with Tracery. This lesson explains how to create simple twitterbots using Tracery and the Cheap Boss Done Quick service. Tracery exists in multiple languages and can be integrated into websites, games, bots, etc.

Using Tracery in Python
Sun 31 March 2019 • Technical • programming • procedural generation
~9 min read
This post is aimed at an intermediate programmer. If you have any feedback, hit me up at @BrettWitty on Twitter.

Introduction
Procedural text generation is about creating interesting text at random. You can use it to spice up user interactions, or make it a core creative input to a game.

In Python you can do this straight out of the box with the random library and string interpolation (formatting):

```python
import random
colour = ['red', 'green', 'blue']
colour_choice = random.choice(colour)

# Using string formatting
text = "The (c) ball bounced down the hall\n\n colour\n colour_choice = random.choice(colour)"

# Using Python 3.6+ format-strings
text = f"The (colour_choice) ball bounced down the hall."
```

Creating Your Own Whimsical Twitter Bot With Tracery

August 15th 2018

Inspired by a series of awesome Twitter bots that tweet seemingly random but hilarious messages such as BoredOMask or thinkpiecebot, I set out to create some of my own. One that tweets random ideas for board games, and another that tweets random spells for Dungeons & Dragons.

Once Upon A Time Stories

7.2: Context-Free Grammar with Tracery • Programming with Text
Kinds of outreach

- Go where people are
- Respect their time and interest
Kinds of outreach

- Evangelist
- Tinker
Useful things to have

- A place where people can talk
- A way for people to talk to you
  - A way for *non-technical* people to contact you
- A known hashtag
Useful things to have

- A site where people can put up art
- Or share source code (and link to remixes)
- Or annotate and curate lists
Useful things to have

- A site where people can put up art
- Or share source code (and link to remixes)
- Or annotate and curate lists

scratch.com
Useful things to have

- A site where people can put up art
- Or share source code (and link to remixes)
- Or annotate and curate lists

https://cheapbotsdonequick.com/source/katesbot
Useful things to have

- A site where people can put up art
- Or share source code (and link to remixes)
- Or annotate and curate lists

botwiki.org
Why do people need a place to gather?

- A place to look for answers
  - (“has this been ported to LISP?”, “is there any French documentation?”, “how do I ...”)
- A place to invest in things *you* care about
  - (kinds of use cases, types of users, languages and nationalities, fandoms, fields of study)
- “Off-label” uses
- Missing community features
What do we owe artists?

- Backwards compatibility
- Documenting broken compatibility
- Not breaking links
- Human-readable formats
- “Forever” museum-quality versions
What do we owe artists?

- Backwards compatibility
- Documenting broken compatibility
- Not breaking links
- Human-readable formats
- “Forever” museum-quality versions
What do we owe artists?

- A file is a file.....right?

- Can you download a copy from the hosting service? (not in Mario Maker)

- Can you edit the file in someone else’s software? (not Photoshop files)

- Can you, a human, read these files? Could you write your own editor? (not for MS Word docs)

- Can someone else build code to run these files?

- Would you feel safe running strangers’ files on your system?
Part I

Language features
How does adoption happen

- You make a good tool
- Everyone likes it
How does adoption happen

- You make a good tool
- Someone is looking for a tool like that at that time
- They have enough time to try it out
- Nothing breaks immediately
How does adoption happen

- You make a good tool
- Someone is looking for a tool like that at that time
- They have enough time to try it out
- Nothing breaks immediately
How does adoption happen

Lessons:

● You aren’t ready when it happens

● “not done yet” won’t stop it from happening

● but “email me for the current version” will
Everyone is too busy

What is the absolute minimum time needed for someone to prove to themselves that your product works?

- journalist
- CTO / tech worker
- high-ranking professor
- new student
Everyone is too busy

What is the absolute minimum time needed for someone to prove to themselves that your product works?

- journalist
- CTO / tech worker
- high-ranking professor
- new student
Everyone is too busy

What is the absolute minimum time needed for someone to prove to themselves that your product works?

- journalist
- CTO / tech worker
- high-ranking professor
- new student
Everyone is too busy

What is the absolute minimum time needed for someone to embed your tool?

What are the dependencies?

Does this require a package manager? A conda container? A server?

Each additional complexity divides your users by 10
Language principles

What is the absolute minimum time needed for someone to embed your tool?

What are the dependencies?

Does this require a package manager? A conda container? A server?

Each additional complexity divides your users by 10
Make 1000 things, fast and bad

What is this new system for?
You don’t know!

Guess from old genres

You can make one prototype
(it’ll be wrong)

The 10th will tell you more

Diversity in users creates diversity in creativity and use cases
Language features for fast creation

- Scaffolded
- Unfoldable
- Little
Language features for fast creation

- Scaffolded
- Unfoldable
- Little

This is my webpage

This is <b>my</b> webpage

This is <b>my</b>  <a href='webpage'>webpage</a>

This is <b>my</b>  <a style='color:blue;font-size:150%;' href='webpage'>webpage</a>

This is <span id='me_1' class='emph'>my</span>  <a class='big_link' href='webpage'>
Language features for fast creation

- Scaffolded
- Unfoldable
- Little

Regular Expression E-mail Matching Example

```
/\w._%+-@\w.-\.[a-zA-Z]{2,}/
```

- Match anything contained within brackets
- Match as many times as possible
- Match the @ symbol
- Match upper and lower case A through Z
- Match a single period
- Match ., _, %, +, and - if found
- Match any character A-Z upper or lower case and any number 0 to 9
- Match at least one more if found
What didn’t work?
What needs more work?
What needs more work?
How do we predict user needs?
How do predict user needs?

A metagrammar generates new art-generating grammars

- shape: ["circle", "triangle", "rect"...]
- Color: ["blue", "magenta", "grey"...]
- ArtGrammar: ["...SetColor: #Color#
  MakeShape: #someShapeType#..."]

A new bot is made by combining an art-generating grammar and a generated or authored heuristic

Many possible (arbitrary) heuristics

- likes: pink (% of red or pink pixels in the art)
- likes: full canvas (100 - % white pixels)
- likes: many round things (# of times "circle" or "ellipse" appears in svg output)

Make and evaluate art
How do predict user needs?
How do predict user needs?

Lessons:
Users may discover features
Users may come up with terrible workarounds
Empty spaces will get filled
Design for surprise and flexibility
How does modding work?
Lessons from Tracery/Chancery

- Users will define what the tool can make
- Design for openness:
  - What can happen with the tool
  - Where the tool can be embedded (in 2045!)
  - Where source files come from, and where they go
- THERE IS NO TIME!
  - Respect user time, and user attention, and user self-esteem