TerseTalk

An Android App for Programming Recreational Maths

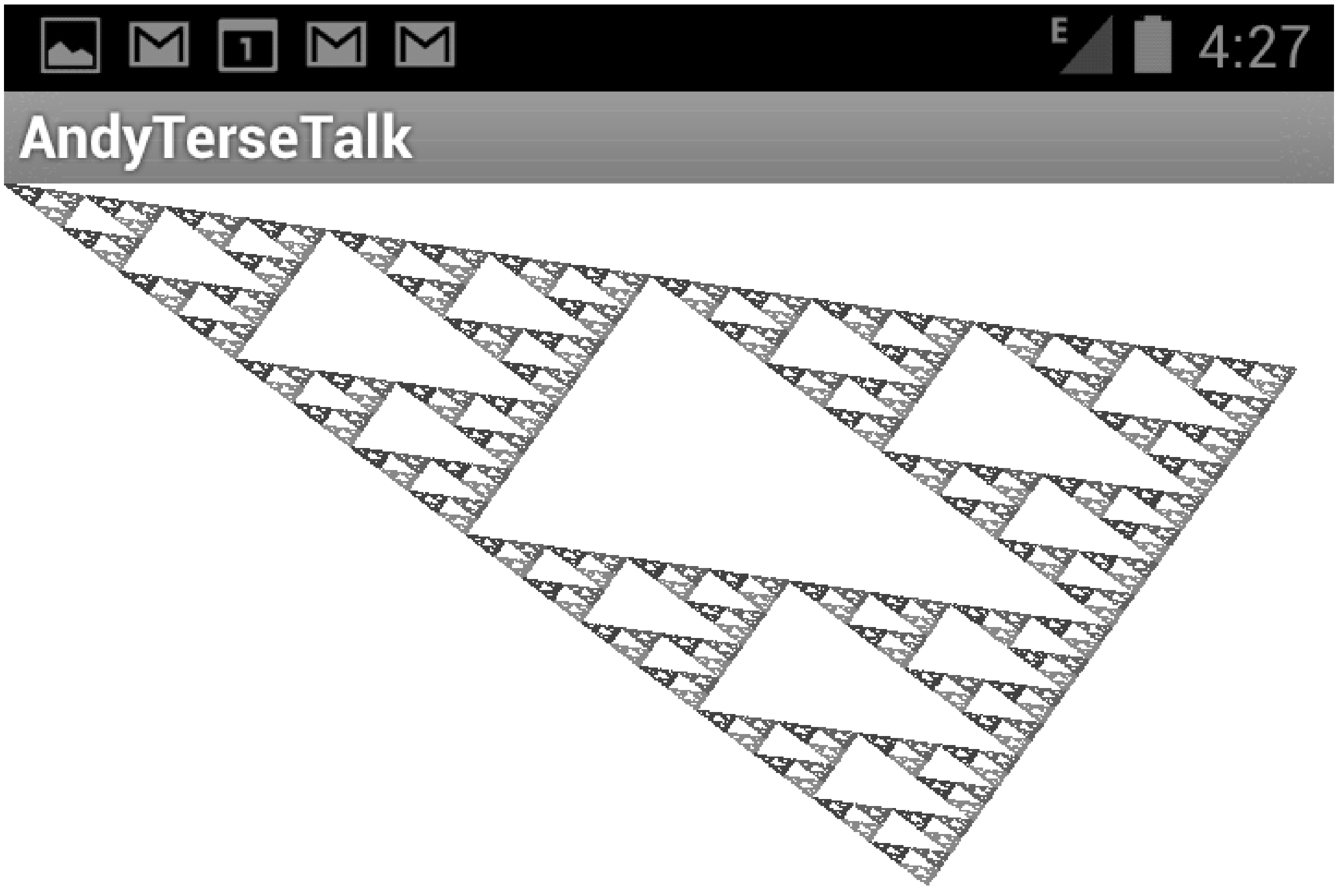
Strick (Henry Strickland) <strick@yak.net> April 2012

http://tersetalk.yak.net/

I program my home computer.
Beam myself into the future.
– Kraftwerk (1981)

TerseTalk is my gift to G4G10.

- It's a programming language that I created.
- It's an android app for phones or tablets.
- It's a complete environment for developing programs and drawing graphics.
- It's inspired by the Smalltalk-80 programming language and environment.
- It's free and open source.
- It's designed to be easy to type on a default Android on-screen keyboard.
- It's designed for experimenting with Gardner's style of Recreational Mathematics.



As a demo, I'll show how to draw Sierpinski's Gasket using the Chaos Game & TerseTalk. (Wikipedia has articles on the following theorem and game and gasket.)

Barnsley's Collage Theorem: If you can completely cover a target T with tiles T_i that are each affine transforms of T, then the functions f_i that map T to each T_i form an Iterated Function System (IFS) whose "strange attractor" is the target.

Barnsley's Chaos Game: Pick a starting point p_0. Generate the sequence of points p_0, p_1, p_2, … by iterating the IFS on p_0, randomly picking one of its member functions at each step, and applying it to the previous point to make the next point. So if the member functions are f_1, f_2, and f_3, and you make random choices 2, 1, 3, 2, …, your sequence will be

$$p_0, \quad f_2(p_0), \quad f_1(f_2(p_0)), \quad f_3(f_1(f_2(p_0))), \quad f_2(f_3(f_1(f_2(p_0)))), \quad \ldots$$

This sequence will converge on the "strange attractor" of the IFS. You might discard the first few points, which might not be near the attractor yet. But plot a finite head of the rest of the sequence, and you'll get an approximate image of the target of the IFS.

Sierpinski's Gasket lives on the plane between three corner points C_1, C_2, and C_3. It's trivially tiled with 3 shunken copies of itself: the three largest sub-triangles T_1, T_2, and T_3, that contain the corners C_1, C_2, and C_3. The affine transform f_i from the entire gasket T to each tile T_i is simply the function "go halfway from where you are, to corner C_i".

On the following page is the TerseTalk code for a "draw" method that does this. It starts with the origin (0, 0), which happens to be the first corner, and then iterates a step of picking a corner at random and going halfway to that corner. Each point is plotted for 50,000 iterations. So you can see the Gasket emerge as points are individually drawn, the intermediate plot is "posted" to the screen every 200 iterations. It takes around 10 seconds to run on a fast (Samsung Galaxy Nexus) android phone.

Exercises for you to try on your android:

- What do you get if you have more than 3 corners?
- Devise a way to color the gasket, based on the random numbers chosen. (If you look carefuly at the previous page, there are 3 shades of gray, based on the random number chosen 4 steps ago.)
- What happens if your probability distrubtion is not 1/3, 1/3, and 1/3?
- Draw ferns and trees. How can we draw the Menger Sponge (which is 3D)?
- Write a TerseTalk app for the iPad, and convince Apple to allow it.

Visit `http://tersetalk.yak.net/` to find the latest on how to create classes and methods for graphical programming in TerseTalk. Also I plan to have a mechanism for sharing your creations with others.

TerseTalk code sample: Statements end with periods. Comments are in “double quotes”.
All values are objects, instances of some class, which is represented by a class object.
All work is done by sending messages to objects. The name of the message follows the receiving object.
For instance, “Num rand: 3” sends the “rand:” message to the class object Num with argument “3”.
Some messages (“unary”) take no argument, like “self white”, which returns our copy of white ink.
Syntactically, unary messages bind tightest, then binary operators like “+” and “/”, then messages with arguments (like “rand:” and “dot:”), then the list-constructing commas and semicolons.

```
    "Draw Sierpinski's Gasket."
```

“0,0” constructs a list with two elements. The variables in the list “x,y=” are assigned successive elements.

```
    x,y= 0,0.
```

These are the corners of the triange. The commas construct lists of 2 elements (x, y point values), which are syntactically nested inside the outer list of 3 elements, construted by the semicolons.

```
    cs= 0,0; 500,380; 700,100.
```

The DO loop is executed 50000 times. The index variable “i” will range from 0 to 49999.
(All indexing is “0”-based).

```
    FOR(i : 50000) DO(
```

The class object for “Num” accepts the “rand:” message, and returns a random integer from 0 to 2.

```
        r= Num rand: 3.
```

“at:” indexes into the list “cs” of 3 corners, to pick a random corner, and its two elements are stored in the two variables “cx” and “cy”.

```
        cx,cy= cs at: r.
```

These two lines are the function “go halfway to the chosen corner”, by averaging the x and y coordinates of the current point x,y with the corner cx,cy. The division operator “/” is floating point.

```
        x= (x+cx) / 2.
        y= (y+cy) / 2.
```

As a convenience, “self white” returns an object of white Ink.
The Ink is used to draw a single dot on the (by default, black) Screen.

```
        self white dot: (x,y).
```

So you can see the Strange Attractor slowly emerge as it is being drawn, we “post” a copy of the current Screen object every 200 iterations. “%” is the modulus operator.

```
        IF(i % 200 == 100)
                THEN(self post).
```

TerseTalk is my own invention. The basic syntax for messages and operators, and the class and object structure, are taken from Smalltalk-80. On top of that, I added “,” and “;” for constructing lists. Then I added a new syntax I call “macros”, for the `FOR()DO()` and `IF()THEN()` structures. There are other syntactic shortcuts to make it easier to type on Android phones & tablets. See the web site for more.

```
    ).
    "http://tersetalk.yak.net/"
```