Drawing by the rules

with Context Free (www.contextfreeart.org)
by Dr Kaz Grace of the University of Sydney (kazjon@arch.usyd.edu.au).

What do we mean by “rules”?
• Formally, a “rule” converts a non-terminal symbol into a set of other symbols.
• A rule can call other rules and transform how those rules are applied.
• Rules don’t know anything about how, when or where they were called, just the current transformation.

Getting Started
• Rules: tell ContextFree what to draw.
• Adjustments: change how rules are drawn.
• The three base rules: SQUARE, TRIANGLE, CIRCLE.

Try this:
startshape SHAPES
rule SHAPES {
   SQUARE{}
   TRIANGLE{x 1.5}
   CIRCLE{x 3}
}

Exercise: Can you change this so the shapes go (from left to right) square, circle, triangle?

Rules can call other rules

Try this:
startshape SHAPEGROUPS
rule SHAPEGROUPS {
   SHAPES{}
   SHAPES{y -2}
   SHAPES{y -4}
}
rule SHAPES {
   SQUARE{}
   TRIANGLE{x 1.5}
   CIRCLE{x 3}
}

“SHAPEGROUPS” is a rule that calls the rule “SHAPES” three times, but SHAPEGROUPS itself does not do any drawing.

Exercise: What would this look like if the “y -2” and “y -4” adjustments were taken out? Sketch your guess in the space below, then try it and see!
Stacking up adjustments
• Adjustments can be applied any time a rule is drawn.
• Adjustments always stack. That means each rule is affected by all the adjustments applied to the rule that called it. Adjustments even affect other adjustments.

Try this, making sure to keep the “SHAPES” rule from before:
startshape ADJUSTEDSHAPES
rule ADJUSTEDSHAPES {
   SHAPES {rotate 10}
   SHAPES {y -2 x -3}
   SHAPES {y -4 size 2}
}

Notice how the size adjustment scales not just the shapes, but the space between them!

Exercise: Can you change this so that the three triangles are touching? If you’re not sure, start changing numbers and see what happens!

Looping with adjustments
• Use “x * {adjustments} RULE{}” to call a rule multiple times applying the adjustment incrementally.

Try this:
startshape SQUARES
rule SQUARES {
   10 * {x 1.5} SQUARE{rotate 5}
   10 * {x 1.5 rotate 5} SQUARE{y 2}
}

Exercise: Note the difference between the two lines of squares in the above example. Write out your explanation of why this happens below:

List of adjustments
• “x <num>” moves on the x axis by num.
• “y <num>” moves on the y axis by num.
• “size <num>” (or “s”) scales in both axes by num.
• “size <num1> <num2>” (or “s”) scales in the x axis by num1 and the y axis by num2.
• “rotate <num>” (or “r”) rotates by num degrees.
• “flip <num>” (or “f”) reflects about a line at num degrees.
• “hue <num>” (or “h”) changes the hue by num degrees. Hue is colour, and ranges from 0 to 360. It starts at 0.
• “saturation <num>” (or “sat”) changes the saturation by num. Saturation is the intensity of the colour, and ranges from 0 to 1, but starts at 0.
• “brightness <num>” (or “b”) changes the brightness by num. Brightness is the lightness of the colour, and ranges from 0 to 1, but starts at 0.
• “alpha <num>” (or “a”) changes the opacity by num. Alpha ranges from 0 to 1, but starts at 1.
• saturation, brightness and alpha increase the current value by num% if you give them a positive number, or decrease it by num% if you give them a negative one.

Exercise: You now know all the different commands in ContextFree. You can now draw millions of shapes! Draw 100 different triangles. Every triangle must be unique in some way, and the more different they all are, the better!

Start with something like this and add adjustments:
rule TRIANGLEGRID {
    10 * {y 1.5} TRIANGLEROW{
rule TRIANGLEROW {
    10 * {x 1.5} TRIANGLE{}
}
}

END OF PART 1
Rules may call themselves
• What happens if you call a rule inside itself?

Try:
startshape FADING_CIRCLE
rule FADING_CIRCLE {
  CIRCLE{brightness 0}
  FADING_CIRCLE{brightness 0.25 size 0.75}
}

• ContextFree will keep going unless the shapes it draws are smaller than 1 pixel across.
• When you do this, you really should include a size command to make it smaller each time, or it won’t ever stop!

Recursion
• Recursion is when part of a program calls itself over and over until there’s some reason to stop.
• Recursion is very cool, both for drawing things and for some quite complicated programming tasks.
• Recursion is very hard. Tons of university students have trouble with this every year, so don’t worry if it doesn’t make sense at first. Try messing around with different things in Context Free until you get a good idea what’s happening! You might just make something that looks great along the way!

Here’s another example:
startshape SPIRAL
rule SPIRAL {
  20 * {y 0.5 s 0.975 sat 0.2 b 0.1} SQUARE{}
  SPIRAL {x 1 rotate 3 size 0.99 hue 3}
}

Randomly chosen rules
• If there are two copies of a rule, ContextFree picks one to run at random each time it’s called.

Try this:
startshape RING
rule RING {
  12 * {x 1 r 30 hue 30} FADING{sat 0.5}
}
rule FADING 1 {
  CIRCLE{b 0} FADING{b 0.2 s 0.9}
}
rule FADING 0.25 {
  SQUARE{b 0} FADING{b 0.2 size 0.9}
}

The two numbers “1” and “0.25” are the chance each duplicate version of the rule will be chosen. In this case the first “FADING” is four times as likely to run as the second. If you’re not quite sure what’s going on here, try changing the startshape to “FADING”.
Exercise: Make a third fading rule which does something completely different! The weirder looking the result, the better!

Randomness and recursion
• A rule can be used to sometimes add adjustments.

Try this:
startshape SPIRAL
background{b -1}

rule SPIRAL {
  10 * {s 0.75 sat 0.25 b 0.25}
  SQUARE{}
  SPIRAL {x 1 rotate 5 size 0.95 hue 10}
}
rule SPIRAL 0.25 {
  SPIRAL {flip 0}
}

That “background” command just makes the background black.

Exercise: Explain in your own words what the “flip” adjustment is doing here. How would you make the “flip” effect less likely to happen?

Splitting while recursing
• What if a recursive rule variant calls itself twice?

Let’s find out:
startshape SPIRAL
background{b -1}

rule SPIRAL 0.99 {
  10 * {s 0.75 sat 0.25 b 0.25}
  SQUARE{}
  SPIRAL {x 1 rotate 3 size 0.99 hue 3}
}
rule SPIRAL 0.01 {
  SPIRAL{}
  SPIRAL {flip 0}
}

Note how rarely this rule will get called. The second rule is called just 1% of the time! Experiment with the chance that this second rule will fire. It’s a good thing Context Free lets you stop a render in progress!

Recursion and randomness allow you to make a single grammar that draws many different shapes. You can write just one set of rules that generates millions of different complex drawings.

Exercise: I want you to invent a plant. Any kind of plant is OK - flower, moss, bush, tree, etc. But not just an ordinary plant, an ALIEN plant. One like nothing on Earth. As different as possible,
if you please! Every time you render it should draw a different plant of the same species. Write a description of your plant species and where it grows below, and submit 3 different photos of interesting specimens!