Doodle Explore GSoC Proposal

Mikail Khan
Scala Center

Synopsis & Motivation

Doodle is a generative art library. Using it, programmers can create artwork, statistical model visualizations, or other graphics. These graphics are generally parameterized by variables such as line width or color. The goal of Explore is to provide an eDSL for describing parameters and their constraints such that a UI for exploring the parameter space can be generated.

These parameters can affect the appearance of the art in several ways. Line width or fill color, for example, can have a direct effect on appearance. However, there may be other, less direct parameters in a visualization such as a random seed, distance metric, or regularization term in a clustering algorithm. These parameters and their constraints define a parameter space, and the visualization algorithm is a map from a point in the parameter space to a graphical output.

Generating a simple UI to explore the parameter space can be useful in many ways.

For example:

- A generative artist can quickly find the most interesting settings
- A teacher could demonstrate the effects of different parameters
- A statistician can better understand variable effects on a given model

In the past Doodle had an explore package. However, it depended on packages that were not ported to Scala 3, and it generates the UI completely from type information. While convenient, this is fundamentally limiting; a parameter could be constrained to the Int type, for example, but it could not be constrained to a specific range. Implementing Explore as an eDSL enables this and opens up some further options

Deliverables

The core deliverables are:

- the Explore eDSL framework
- the Java2D frontend
- documentation

Optional deliverables are:

- the HTML frontend
- type-directed construction

The Explore eDSL framework

At its core, the Explore eDSL is a language for constraining parameters. Parameters in a generative art piece can be constrained in many ways, including by type or range. For example, the stroke width of a piece may be constrained to the Integer type in the range of 0 to 5. This might be written with Explore as below to produce a slider in the UI:

```
Explore.int.within(0, 5)
```

Other possible constraints include:

- Constraining an option to a list of string values or enum members in a dropdown
- Setting a parameter to an image texture through a file selection menu

There are infinite possible types of constraints, so the Explore DSL should be extensible. Additionally multiple frontends such as Java Swing or a web page should be supported. A tagless final implementation of the eDSL lends itself nicely to solving these problems.

The Java2D frontend

Java2D is a well supported framework and a reasonable default. As a retained mode GUI, Java2D rendering (initialization and updates) are side-effects.

The HTML / web browser frontend

A browser frontend is appealing for its portability and customizability. Unlike Java2D, an HTML frontend could be initialized as an entirely pure value.

Type-directed construction

While the previous Explore package suffered from type-directed construction due to its lack of flexibility, it will still be quicker than using the full DSL in some situations.

Timeline

- May 20 - June 12: Community Bonding Period, familiarize with the community, codebase, and contribution guidelines.
- June 13 - July 1: Implement the core of the eDSL, get a single type of constraint (integers and ranges) working with a basic Java2D frontend
- July 4 - July 15: Implement further constraints and menus such as for enum members, images, colors, etc.
- July 18 - July 22: Focus on documentation, including DSL features, how to extend the DSL, how to add frontends.
- July 25 - July 29: (Bonus) Add an HTML web frontend
- August 1 - August 12: (Bonus) Further refine, improving code documentation and written documentation, consider adding type-directed construction

Personal Details / Related Work

Name: Mikail Khan
University: Purdue University
Email: mikail@mikail-khan.com
Phone: 703-689-1353
Timezone: EST
Website: https://mikail-khan.com
GitHub: https://github.com/mkhan45

I am a sophomore majoring in Computer Science at Purdue University with a focus in programming languages and compilers. I will be free from May 9th to August 15th, through which I will be able to work up to 40 hours a week.

My knowledge of Scala is mostly based on my work in Purdue's compilers class, which is essentially the same as EPFL's Advanced Compiler Construction course. I will also be conducting research at Purdue working on a Scala codebase, through which I will become proficient enough in Scala to complete this project. The research itself concerns partial evaluation and writing a Datalog interpreter using Professor Rompf's LMS framework.

My other relevant experience includes a fair bit of work with writing interpreters in Rust, Haskell, and OCaml. Some examples are my science/engineering DSL, CalcuLaTeX (https://calcula.tech, written in Rust), and my functional programming language RustScript (https://github.com/mkhan45/RustScript2, written in OCaml).

Another relevant project of mine is SIMple Physics, a set of simulators to help high school teachers and students graphically demonstrate different physical concepts. A core feature of each simulator is its scripting integration; using Rhai (https://github.com/rhaiscript/rhai), students and teachers can create labs, experiments, or even write their own physical laws. I have not finished writing it yet, but you can find the beginning of the tutorial here https://simple-physics.org/tutorial/gravity.html, and a demo here: https://gravity.simple-physics.org/. The basic update and scripted gravity are the most interesting.