Processing.R

Ce Gao, gaocegege@hotmail.com

Project Abstract

The goal of this project is to offer a new mode in Processing for R Language, which allows users to write Processing sketches in R.

Project Description

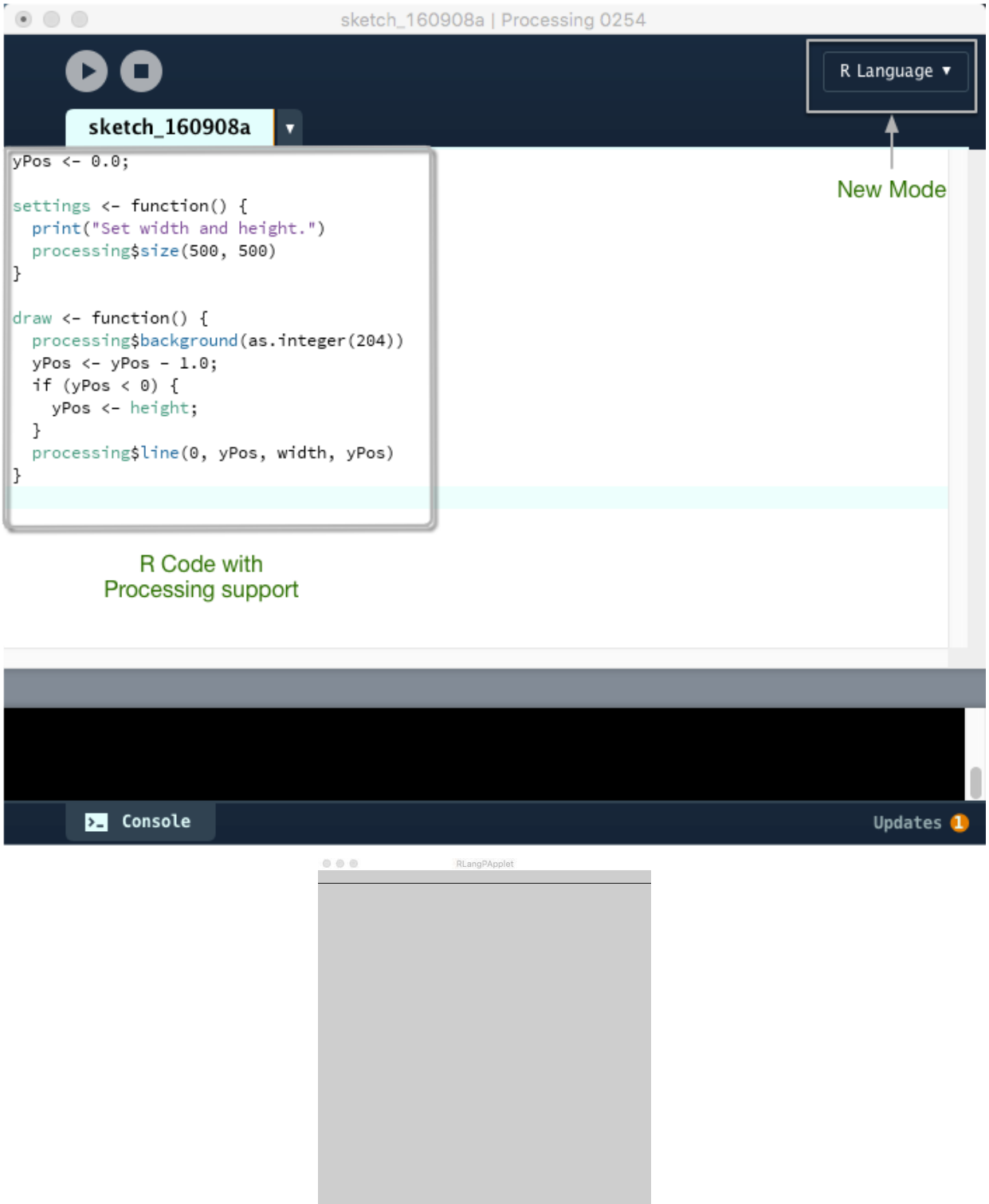
Motivation

Now Processing has two language modes: Java and Python. And p5.js ports core concept of Processing on native Javascript. Java, Python and Javascript are among the most popular and widely used programming languages in the world, but these languages have similar audiences.

R is an open source programming language and software environment for statistical computing and graphics. It ranks 15th in TIOBE Index for February 2017. Compared to Java or Python, R is concentrated on statistical computing although it is designed to be general purpose. There's a demand for data visualization and visual art in R. Processing.R is to offer a new mode in Processing for R. It is a good way to expand the user base for Processing and reduce learning curve of Processing for people using R.

Prototype

I have built a prototype for this proposal: Processing.R. It has basic abilities to write R scripts using the functionality of Processing, but it is incomplete. There are some key features to implement. In GSoC 2017, I propose to fix critical bugs and add some important features to make the production ready.



Processing.R Prototype

Implementation

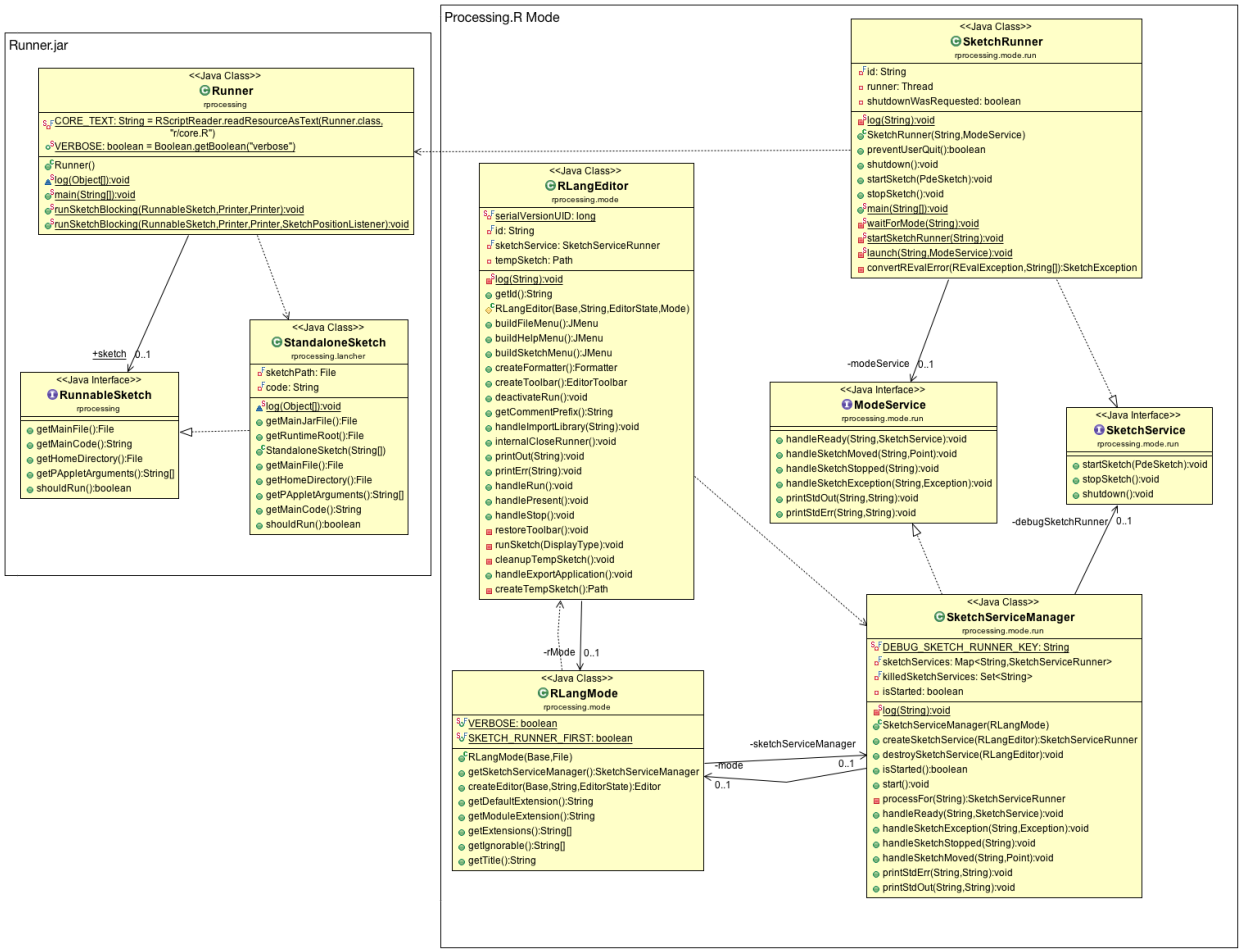
Now Processing.R has two ways to use:

1. A jar “Runner.jar” is offered, which allows users to have a try without the installation of Processing app.
2. A mode in Processing app, which is used to write and run code in the IDE.

The UML class diagram shows main classes in Processing.R.

Runner.jar has two main components: Standalone Sketch and Runner. Standalone Sketch represents the sketchbook. Runner inserts some built-in functions and variables into R global scope; interpret code by renjin, which is a JVM-based interpreter for the R language; and create PApplet to run sketch code.

Processing.R Mode extends Editor and Mode classes to support Processing app. RLangMode has a Sketch Service Manager, it creates a new Sketch Service and Sketch Service runs a Sketch Runner to run specific logic.



Development Process

Community Bonding (May 5 - May 30)

The aim in Community Bonding is to learn about Processing community, dive into the architecture of language modes in Processing, set up a development environment for coding and debugging, and optimize the process of CI for Processing.R.

Now Processing.R builds a jar in CI process. In Community Bonding it is better to build all targets in build.xml.

Coding Phase 1 (May 30 - June 26)

There are 4 weeks in Coding Phase 1 and an evaluation after the phase.

The aim in Coding Phase 1 is to fix critical known bugs and support more built-in functions. There are some bugs in the prototype, one of them is critical.

Besides this, Processing.R only supports a subset of all Processing built-in functions. There are more than 250 functions in Processing and 5 of these functions are supported in Processing.R. In this phase functions about shape, 2D Primitives, environment, curves and vertex should be ready.

Week 1 (May 30 - June 5)

The users could not input new lines in R mode now. The aim in week 1 is to fix the critical bug in Processing.R. Formatter is the possible cause to this bug but it needs to be confirmed with further analyses.

Week 2 (June 6 - June 12)

The aim in week 2 is to support functions about shape, 2D Primitives and curves. There may be some tricks in the implementation because of the different type systems between Java and R. Processing.R is usable to some use cases after this week.

Week 3 (June 13 - June 19)

The aim in week 3 is to support functions about environment and vertex, similar to week 2.

Week 4 (June 20 - June 26)

In week 4, The goal of this phase has been attained and it's better to review the code and fix some known bugs in the previous code. And the code should have corresponding documentation. Examples can be added about newly supported built-in functions.

Coding Phase 2 (June 27 - July 24)

There are 4 weeks in Coding Phase 2 and an evaluation after the phase.

The aim in phase 2 is to support some CRAN and BioConductor R libraries, improve code quality according to automated code reviewing tool(codacy), and add unit test cases for Processing.R.

Week 5 (June 27 - July 3)

The aim in week 5 is to support R libraries in Processing.R.

There are some differences between the way Renjin manages packages compared to the way that GNU R manages packages, This can be done in multiple ways, where the clearest way would be to install packages in runtime. If that is not possible, the other way is to install packages ahead of time.

Week 6 (July 4 - July 10)

The aim in week 6 is to improve code quality according to codacy. Maintainability is important to open source projects. Now Processing.R has some problems in code complexity and error-prone, these problems should be solved in this week.

Week 7 (July 11 - July 17)

In week 7, there are test to be done. The main logic in PApplet, Sketch and other main components should have corresponding unit test cases.

Week 8 (July 17 - July 24)

In week 8, again, a pause for testing and review of code and updating documentation will be a good idea. Besides this, CI for Windows could be added into the project. AppVeyor is a helpful tool to implement this task.

Coding Phase 3 (July 25 - Aug 29)

There are 4 weeks in Coding Phase 2 and the final evaluation after the phase.

The aim in this phase is to solve the problem about RLangPApplet, add Mix mode, and remove `processing` prefix in built-in function calls.

Week 9 (July 25 - July 31)

The aim in week 9 is to solve the problem about RLangPApplet singleton. RLangPApplet is created every time the user clicks `run`.

Week 10 (Aug 1 - Aug 7)

In week 10, The work about active/mix/static modes is in the plan. The mode is a syntactic sugar, it is designed to make code easier to read or to express. Processing.R has some problems in static mode now, because there is no check in AST level. Finding and evaluating the function calls such as size and others that should be used in initialization of the PApplet before running is a possible solution.

Week 11 (Aug 8 - Aug 14)

The aim in week 11 is to remove `processing` prefix in built-in function calls. It also is a feature to make Processing.R “sweeter” for human use. The built-in functions now have a prefix “processing”, because there are some functions in R with the same name as in Processing. @GoToloop gives me a idea to implement a pre-processor, so Processing.R scripts may not be valid R scripts. We have a discussion about the issue.

Week 12 (Aug 14 - Aug 20)

It is the final week, code reviewing and bug fixing are the main tasks in this week. There are some preparations and cleanups to transfer the ownership to Processing Foundation.

Buffer Time (9 days)

There are 9 buffer days in case something didn’t go as planned in the weeks before.

More about Me

Contributions

Contributions to Processing

- https://github.com/gaocegege/Processing.R
    - Prototype for this proposal

Contributions to Other Open Source Projects

- github.com/caicloud/cyclone/commits/master?author=gaocegege
    - Its goal is to offer a hackable, lightweight CI/CD platform built on Docker.
    - My work includes YAML parser which parses YAML configuration file to container configuration, container runner which starts Docker containers to run CI/CD tasks, and event manager which processes events asynchronously and redirect requests to corresponding components.
    - Collaborator of this project.
- github.com/coala/coala-vs-code/commits/master?author=gaocegege
    - Visual Studio Code plugin for coala.
    - Author of this project.
- github.com/docker/docker/commits/master?author=gaocegege
    - Add a command: docker service ps -q.
    - Have been released in docker 1.13.1 (2017-02-08)
- github.com/opencontainers/runc/commits/master?author=gaocegege
    - Fix a bug related to docker.

Personal Projects

- github.com/gaocegege/scrala
    - Scala crawler(spider) framework, inspired by scrapy.
- github.com/gaocegege/flower-power
    - A simple animation written in Processing.
- github.com/gaocegege/dronecraft
    - Drone client in minecraft, inspired by dockercraft.
    - Written in Go and Lua.

About Me

I'm a freshman in a master's program in Shanghai Jiao Tong University, my own research orientation is container-based OS virtualization (COS virtualization), I have contributed to many container-based projects such as docker, kubernetes, runc and so on. I'm a heavy user of GitHub and know the workflow in GitHub.

I had a project about data visualization and started to learning Processing in my second year of university. Learning Processing, Second Edition helped me a lot on self learning. After that, I have implemented some interesting projects in Processing such as flower-power and game-of-life-in-processing. As for R Language, I have earned the certificate for r-programming in coursera, and wrote some blog posts about R in Chinese. Object Oriented Implementation in R is one of the most popular posts and it records one way to write OO code in R.

Processing.R came to me when I was trying to demonstrate the result in R by a variety of charts. I tested some of the existing libraries and found that most of them are not fancy enough. Processing.R is the possible solution to this problem.

Student Info

Name:	Ce Gao
Email:	gaocegege@hotmail.com
Github:	github.com/gaocegege
Time Zone:	UTC+08:00 (China)
Location:	Shanghai, China
Processing Forum ID	gaocegege