GSoC 2017 Proposal to Kivy (Python Software Foundation)
Project: KV Compiler: A compiler for the KV language

Sub-organization information

Kivy

Mentors

Matthew Einhorn (matham)

Student Information

Name: Yash Jain

Email: yashjain.lnm@gmail.com

Telephone: +919414419320 / +919660243960

Time Zone: Jaipur, India UTC+5:30

IRC: yaki29@irc.freenode.net

Source Control Username: http://www.github.com/yaki29

Facebook: https://www.facebook.com/profile.php?id=100003284453761

Blogs: https://yaki29.github.io/Blog (Will be available)

University Information

University: The LNM Institute of Information Technology, Jaipur

Major: Computer Science and Engineering

Current Year: 2nd Year

Expected Graduation Completion: By June 2019

Degree: B-Tech

Project Proposal Information

Proposal Title: KV Compiler: A compiler for the KV language

Proposal Abstract: This Proposal is based on understanding of kv language in the ideas page provided by Kivy Organization. The goal of the project will be to create a compiler which compiles kv code into python code, with debug/optimization option.

Project Description:

A compiler is a computer program (or a set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language). It might include debugging and optimization as well. Here in this case the source language would be kv-lang and the target language would be python.

Currently, the bindings are not at all optimized because upon each widget creation all of the rules are re-evaluated and bound. This process can be significantly optimized by pre-compiling the kv code, especially the bindings. Hence we need a better and optimized compiler.

- **How the compiler will be implemented(Technical Overview):**

What our target is to design a consistent, modular and extensible compiler which compiles kv code into python code.

First, I'll work on already implemented compiler. I have the basic understanding of what it is doing and how the methods are implementing the features, like generating the parse tree, analysing the widget tree etc. It still has conflicts which have to be resolved and without refurbishing all the methods we can't go further. It's not in the stage to handle the changes made in kivy in recent past. I would do it in the "Community Bonding Period". It is a huge project hence, from beginning I'm trying to save more and more time.

After that, I would start working on new basic features which are not yet implemented in the compiler but could be of great use. Today, everyone owns a computer having multiprocessors. We can use algorithms which convert sequential code into multi-threaded or vectorized(or even both) code in order to utilize multiple processors simultaneously in a shared-memory multiprocessor(SMP) machine. This will highly reduce the time of execution and it would be a new feature for kivy. Moreover, we can add some basic functionalities like multiple file compilation, which would be useful for further testing and even make the compiler more handy. I will be in regular contact with the mentor and if he suggests some other changes and functionalities to add I would add those.

That would be like stretching for forthcoming tasks**. Further**, I would go for complex optimization techniques. Work on optimization modules that are going to be useful during the process or those which are suggested by the mentor. This would be heart period of this project. Extensive testing and debugging along with it would be necessary. I would also code the examples which can be added to kivy which tests every aspect of the compiler, so that in future if anyone wants to work on kv-compiler he won't have to write it himself. Documentation would go in parallel to all these steps.

- **Access to required hardware:**

I personally own Linux, Windows and Android. If anything else is needed I will buy it within a week.

- **Dividing the work flow:**

My summer vacations will start from mid-May and will be ending in mid-July, So I have planned my workflow accordingly. I will be dividing my work in 4 parts.

- Phase-1: Community Bonding Period.
- Phase-2: Before Mid-term Evaluation.
- Phase-3: Before gsoc's 2nd Phase Evaluation.
- Phase-4: Further till the end.

Timeline:

Community Bonding Period:

During this period, I would mostly work on alpha stage compiler programmed by matham. I would make it consistent with the existing kivy development branch. Consistency would be my primary motive. I have already made some corrections which at least bring it in testing stage, means you can compile a kivy file using this compiler code.
I would rewrite or renovate the methods of KVCompiler and lang.py so that it can find out the compiled files and execute them.

Before Mid-term Evaluation (May 30 – June 30):

I would start work on adding new features to the compiler which can be useful.

May 30 – June 15(Week 1 and Week 2)

I would start applying different algorithms to the compiler to make it more time efficient.
Like I mentioned earlier, we can take use of multiprocessor computers and apply some algorithms which divides the processes into threads which would highly reduce the time.
Documentation and testing would be done regularly. I would regularly update my blog.
If my mentor wants anything else too, I will start that too.

June 16 – June 30(Week 3 and Week 4)

I would add some basic but important functionalities to the compiler which would be relevant in testing the compiler's working like multiple file compilation, custom files' compilation.
Test the results regularly and make changes in further schedule if needed.
I would be in constant contact with my mentor if he wants some other features to add too, I would add them.
I would update my blog in the end of this period and make preparations for the mid-term evaluation process in parallel.

Before Gsoc's 2nd Phase Evaluation (July 1 – July 28)

Implement complex optimization techniques that are found relevant and necessary.

July 1 – July 15(Week 5 and Week 6)

Phase 1 was like a test for the real exam. During this period I would implement the optimization modules that are going to be useful or that are suggested by the mentor.

July 15 – July 28(Week 7 and Week 8)

This period would be similar to the earlier one. The focus would lean towards the testing part.
Mainly, the whole month would be needed to apply the optimization methods and testing and debugging them as we keep on working.

Further till end (July 30 – August 25)

This duration would be on testing and example writing.

July 30 – August 12 (Week 9 – Week 10)

Extensive testing would be needed. I would compile hundreds of kv files and test if there are any bugs left. This would be different from the regular testing, regular testing is mainly for the features that are added during that period, these tests would cover the whole kivy language.
I'm planning on adding example section as well in the compiler. I would add formalised examples that tests every aspect of the compiler and show you what compiler is doing. This would be a good thing for a newbie who wants to work on compiler.

August 12 – August 23 (Week 11 – Week 12)

This would be just wind up for my project. I would add documentation and update my blog about what I have done so far and how was my experience.
I'll make changes to lang.py if needed.
Resolve the bugs which are still left or generated due to regular changes in kivy.

Week 13

This week would be a buffer for any unpredictable delay.

Onwards

Kivy is my first love. Even if I don't get selected I would never forget it and never stop contributing to it.
I will always be ready to solve any issue regarding KVCompiler, because I have worked so hard for this.

Link to a patch/code sample, preferably one you have submitted to your sub-org :
I have implemented the following:

1. **Facelock using Computer Vision techniques(kivy-garden flower). (Merged)**
2. **BeautifulSoup4 recipe for android. (Waiting for approval)**
3. **Camera example for kivy. (Merged)**
4. **Corrected Basic Documentation mistakes in p4a and kivy.**

Other Commitments:

- Have you applied to any other organization? **No.**
- Do you have any other commitments during the main GSoC time period? **No.**
- Do you have exams or classes that overlap with this period?
  My next semester's classes would start from starting August. I mostly work during night so, I don't think that is going to be any problem. Even if any kind of emergency comes up, I'll take extra load during the weekends.

Why am I apt. for this project:

Familiarity with Kivy and its coding style. I have an extreme interest and curiosity towards coding and have been doing it for past 2 years with renowned languages Python, Java and C. I have been contributing to Kivy and its sister projects for past 5-6 months(mainly Python-for-android and Kivy-Garden). I am working on a 2-pass Assembler at my college and some python projects in Kivy, some of them could be found on my GitHub profile. I reached semi final round of international coding challenge, **Snackdown 2016, Codechef.** I have also earned T-shirts and accessories from **OSFY(Open Source for You)** for my articles. One of them is going to publish in April's edition.

Proposal Improvements:

- It will definitely consist of an optimization/debug option, one more option can be added further which recursively compiles all the files in the working directory, so that user doesn't have to sit and compile all the files of his kivy-project.
- What if user just wants to compile some of the custom kv files, he/she can just enter the file names using a space between them it will identify the multiple files and compile them one by one. That will make the compiler more handy.
- I was thinking of adding an example section. This would consist of examples which implements and explains every aspect of the compiler.