GSoC '20 Project Proposal

Model–View–Controller refactoring for Music Blocks

Basic Details:

Full Name

My name is Chandan Prakash and I am 2nd year C.S.E. undergraduate student at **Indian Institute of Technology Mandi .**

Email and Github Username

Email: b18050@students.iitmandi.ac.in , abchandan11@gmail.com
Github Username : b18050
Website : myportfolio
IRC Username: abchandan11
Medium : medium@b18050

Your First Language

My first language of communication is English. I am also fluent in Hindi.

Location and Timezone

Location: Mandi, Himachal Pradesh , India.
Timezone: UTC +5:30 India Time Zone
I'm very excited to work on this project during the summer and I can surely manage my time and be active when the mentors are available.

Share links, if any, of your previous work on open source projects

In my freshman year I got introduced to open source, Github and web development.Initially, I started with some beginner level projects using Javascript , python (like weather-app, stopwatch, calculator etc.) . I also took part in Hacktoberfest 2019. I feel motivated seeing contribution map full of those green dots , so, I keep pushing my work on my Github profile (b18050). It also helped me improve my skills .

I have started contributing to SugarLabs MusicBlocks repository from early December. This is my first time contributing to such a big open source project. The last three months have been a great learning experience for me. Some projects I worked on:

- **Hackathon Project -** My team was runner-up in Inter-College Hackathon organised by EESL. It was organized by E-cell of our college in which we have to solve a problem on electric mobility. We made a web-app for booking timings for power stations and proposed a solution based on feasibility of customers and distributors both.
  Link to the project : energy-effi. Proj. Final project link: hackathon project

- **Contest Reminder Bot -** This project is an application of web scrapping . This script reminds you about the coming programming contests on various platforms like codechef, codeforces, and much helpful for college students. It is written in python using selenium (previously written using beautifulsoup). More about the project can be found here.
  Github link -   zulip botcintest

- **Weather -app -** In my first year , I made a small project in which I learnt to use weather API. This is a web-app which can tell temperature of a city entered by a user. Link to the web-app : Demo , Source code : github link

- **Covid-19 bot -** Recently, I have made a script which will spread awareness and shows the number of active cases in my country, just a web scrapping thing.

- **Contribution to other orgs.**
  - publiclab/mapknitter/pull [#1132]
  - publiclab/mapknitter/pull [#1133]
  - publiclab/mapknitter/pull [#1135]

RIGHT FIT

Convince us that you will be a good fit for this project, by sharing links to your contribution to Sugar Labs.

I am an active contributor to MusicBlocks project of Sugar Labs for the last four months, and now I am comfortable with the code base of MusicBlocks. I am also familiar with Sugar-desktop and sugarizer. I have also set up Sugar-Desktop on my local machine. I have seen activities of both sugarizer and sugar. In recent days, I have learnt more about different widgets used in

MusicBlocks. I am also learning more about Model-View-Controller model which is the main target of this project. I have been coding in Javascript for more than 1.5 years.

Following are some of my major contributions to **MusicBlocks:**

These are my PRs :

- ❏ [#2106] - animated keyboard in mode widget , which fixed issue sugarlabs/musicblocks/issues/ [!2090]
- ❏ [#2115] - documentation added for planet in README.md , which also fixed issue sugarlabs/musicblocks/issues [!2108]
- ❏ [#2123] - added double bar lines at the end of voice ,which fIxed Issue musicblocks/issues/ [!2119]
- ❏ [#2084] - added beginner label on each block individually which helped to fix issue sugarlabs/musicblocks/issues/ [!2082]
- ❏ [#2089] - added save functionality in beginnerMode which fixed issue sugarlabs/musicblocks/issues/ [!1948]
- ❏ [#2132] - save widget state or data after refresh , which will fix issue sugarlabs/musicblocks/issues/ [!2017]
- ❏ [#2060] - added scrolling in palettes menu , will fix issue sugarlabs/musicblocks/issues/ [!2052]
- ❏ llaske/sugarizer/pull [#571] - improved responsiveness of TankOp activity , attempt to fix issue llaske/sugarizer/issues/ [!566]
- ❏ sugarlabs/musicblocks/pull [#2152] - solved timber widget not opening (fixed a typo) and it starts to run
- ❏ sugarlabs/musicblocks/pull [#2130] - removed animation from loadPlugin.
- ❏ sugarlabs/musicblocks/pull [#2113] - distance labels properly shown.
- ❏ sugarlabs/musicblocks/pull [#2073] - updated broken links in documentaion/README.md which fixed issue sugarlabs/musicblocks/issues/ [!2072]
- ❏ sugarlabs/musicblocks/pull [#2045] - Display Project title when loading from the planet which fixed issue sugarlabs/musicblocks/issues/ [!1976]

- sugarlabs/musicblocks/pull [#1998] - added a distance block which fixed issue sugarlabs/musicblocks/issues/ [!1906]

Issues:

- [!2066] - make block from search by choosing from the list .
- [!2067] - display statistics goes into infinite load animation .
- [!2052] - Enable scrolling in palettes menu .
- [!2126] - Bottom part of widget not visible .
- [!2099] - Title issue in keyboard widget , fixed by [#2102].

Also for all the time contributing to MusicBlocks, I have over 30 pull requests and 45+ commits in the code base of MusicBlocks. I have also learnt new things like Musescore , Lilypond and many more while working on this project and therefore want to contribute more to this.
Link to all my commits:
I have also helped new developers to get familar with the code base.
#Link

Your Motivation

What is your motivation to take part in Google Summer of Code?

I came to know about GSoC from my college seniors in one of the club meetings . I felt very motivated listening to their GSoC stories. Therefore , at that time only, I decided to take part in GSOC . After joining Sugarlabs , I came to know that this community is providing free educational activities to children , that added more energy in me to be a part of a community that is doing something for the welfare of society. I started contributing and was very happy to work with respected Walter Bender. I want to take part in GSoC so that I can also take motivation from me .I will also learn so many new things during GSoC.

Why did you choose Sugar Labs?

Music and Codebase:

Among all the programming languages I am quite comfortable with **javascript** and **python.** Since childhood, I have had a keen interest in music . Here , I am getting a wonderful chance to showcase my skills and that too with something I enjoy doing. I have benefited from open-source software countless times. Now I want to give something back to the community.

Learning software for children:
Sugarlabs is doing a noble job by providing free learning software for children . I really like the idea to use technology to promote collaborative learning by different activities of sugar.

Great Mentors:

I started contributing to MusicBlocks in early december and made my first PR (added a distance block) taking help from documentation .But , at that time I have no knowledge of testing it . I am thankful to Walter Bender , he guided me until it gets merged. He taught me how to test something on your local machine. With his help , I got my first PR merged which means so much to me . Debin Ubbarri also comes to guide me in my PR. At any time , the mentors are ready to help , that's why I feel this is really great. James Cameron also helped me in setting up sugar on my local machine and always answered any query on the mailing list. Ibriam helped me in setting turtleBlocks activity.

Moto of SugarLabs:

I really like the theme 'One Laptop Per Child project' to provide educational opportunities to the children with the help of passionate volunteers. Anyone can help in whatever possible way, You need not have to be from a technical background to be part of this community. It is also accelerating development which is the need of time. Everyone is considered equal and independent ideas are always welcomed .

Why do you want to work on this particular project?

Javascript:

The project needs work that can be completed using javascript . Using Javascript, we can design crazy optical effects, games, UI, create your own dynamic site.The frameworks , libraries of javascript are best suited for working on this project. Model - view - controller can be easily implemented using javascript. I have learnt more about implementing M-V-C using JS which will definitely help getting this project done in time.

Learning opportunities:

As mentioned above, I love to learn new things and apply it to day-to-day life and here I am getting the opportunity to learn Javascript ,working with such a big community under the guidance of helpful mentors . This project will help me gain more skills and sharpen my way to get work done. I am eager to learn new things related to music . Recently , I came to know about Musescore (sheet music) . By learning it , I was able to solve some issues in this project. It gives me immense pleasure to learn and solve an issue. My motivation comes from here.

Work needs to be done:

In GCI, some work has already been done on this , there is much more to be done.I am an active contributor to this project . It is noticed that the M-V-C model needs to get implemented in MusicBlocks. Music Blocks intermixes it program logic to the point where it is getting difficult to maintain. When this project will get complete, this will make codebase friendly for everyone . Beginner developers will find it easier to understand the code and contribute to the project.

Project Details:

What are you making?

My project idea consists of two parts. First , I want to complete the work that remained incomplete during GCI 2019 ,i.e. code refactoring of MusicBlocks. Second , I want to solve bugs i.e. add functionality to different widget, planet view and for this I will be solving issues of MusicBlocks.

Part 1

I am refactoring code of Music Blocks to Model - view - controller . Music Blocks intermixes it program logic to the point where it is getting difficult to maintain. This project is to take a deep dive into the code in order to develop a plan and implement a refactoring along the lines of MVC.This project aims to develop:

- ➢ **Simultaneous development:** After this refactoring process , multiple developers can work simultaneously on the model, controller and views. It will also help beginners to understand the codebase easily and start contributing to this project.
- ➢ **Ease of modification:** Because of the separation of responsibilities, future development or modification is easier.
- ➢ **Multiple views for a model** : Models can have multiple views.
- ➢ **High cohesion:** MVC enables logical grouping of related actions on a controller together. The views for a specific model are also grouped together.

There is some progress on MVC during GCI, but there is much more to be done.

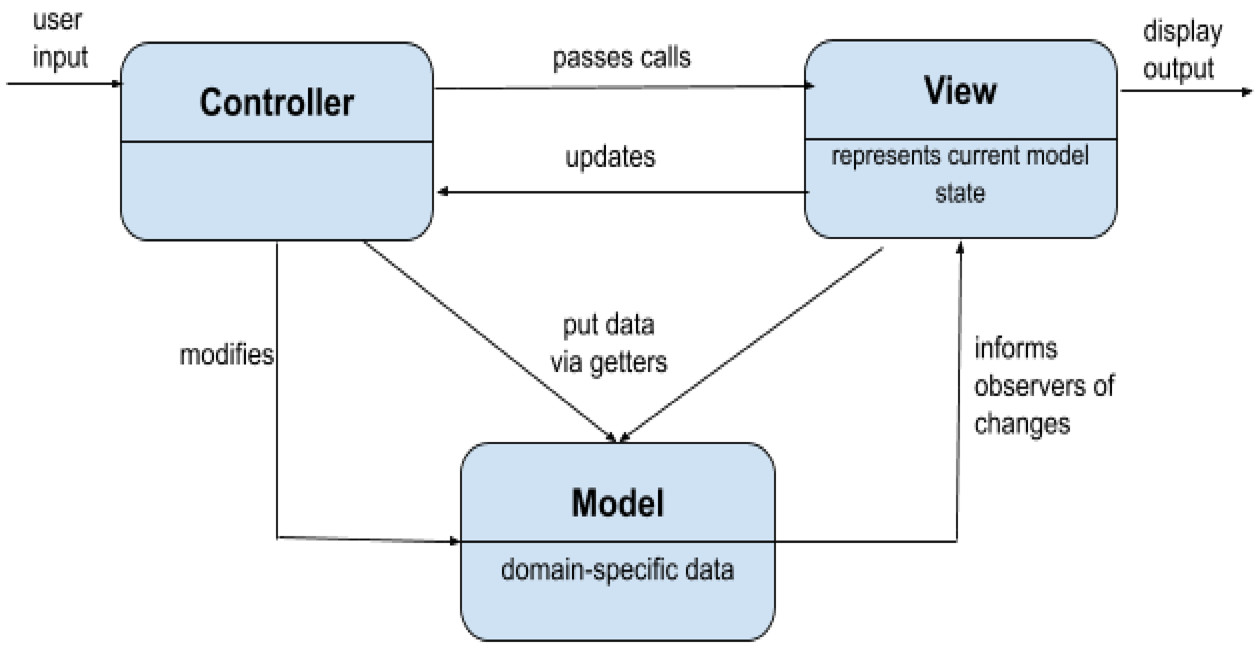
Brief Introduction of M-V-C in this project:

M-V-C stands for model - view - controller. It is an architectural design pattern , not a framework. It separates.an application into three main logical components:

- ❖ **Model** - Basically, it stores data of an application. Music Blocks contains many widgets. When talking specifically about widgets, In this project, for every widget,we will have one model.js file which will store all required data of that widget. Let us take an example of rhythm ruler widget. It stores data in the form of array objects (drums, rulers, dissectHistory etc.). So, it will also contain various methods to access data of the widget.

- **View -** This is what users see and interact with. With respect to widget, it is the widget window that opens when clicking on the widget block . Most of the view had been done in GCI . However, if anything more needs to add , I will be happy to do that. View part of other files like activity.js, blocks.js, logo.js may need some refactoring. I will be doing that as well .

- **Controller -** According to wikipedia, it accepts input from the user and converts it to commands for the model or view. It establishes a relation between model and view, i.e. model and view never interact directly .It is the controller that takes input , processes it to the model , receives data from model and provides the data to view and the user is able to see the output .



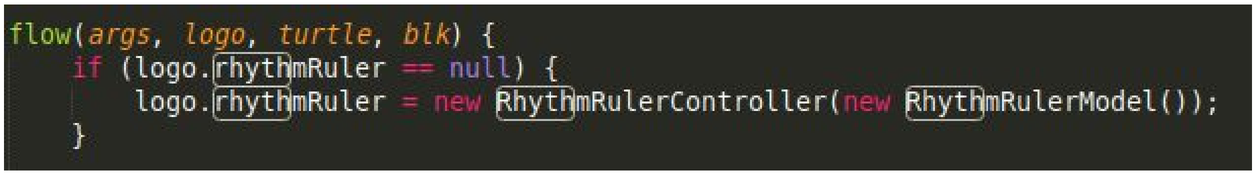
Implementation for widgets

There will be an individual directory for each widget consisting of model and controller js files. The code in **activity.js** also needs refactoring as it is also a mix of model, view and control. So , I will be separating model, view and controller part of them in separate files .Other files such as **blocks.js** , **turtles.js** also need refactoring .

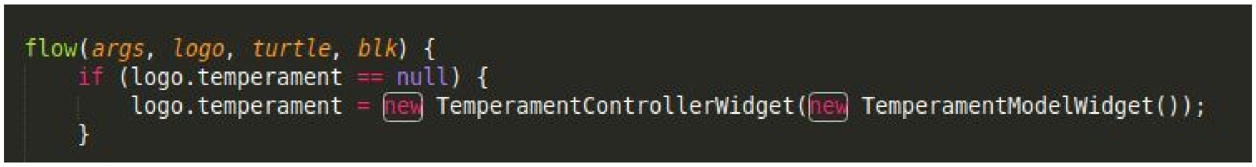
View part was completed during GCI. So, we will be using that in our controller also.

```
var widgetWindow = window.widgetWindows.windowFor(this, “phrase maker”);
```

Now, every widget opens a window when the corresponding block is clicked. This will not change after refactoring , but they will be called differently. For example, **the rhythm ruler** widget will be called like this from logo.js, In the next fig. **temperament widget** is called in the same way in logo.js.

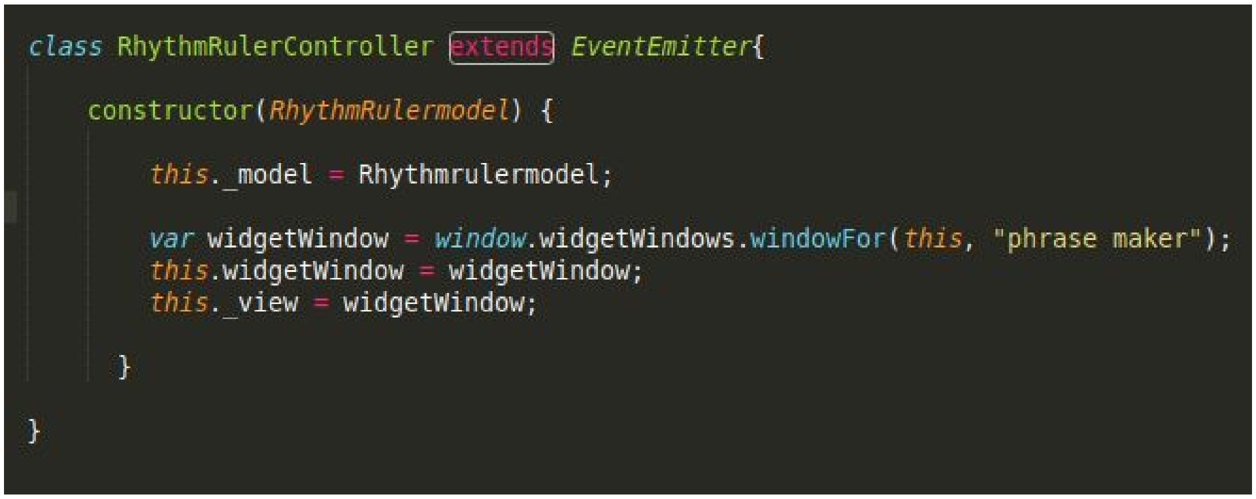


```
flow(args, logo, turtle, blk) {
    if (logo.rhythmRuler == null) {
        logo.rhythmRuler = new RhythmRulerController(new RhythmRulerModel());
    }
```



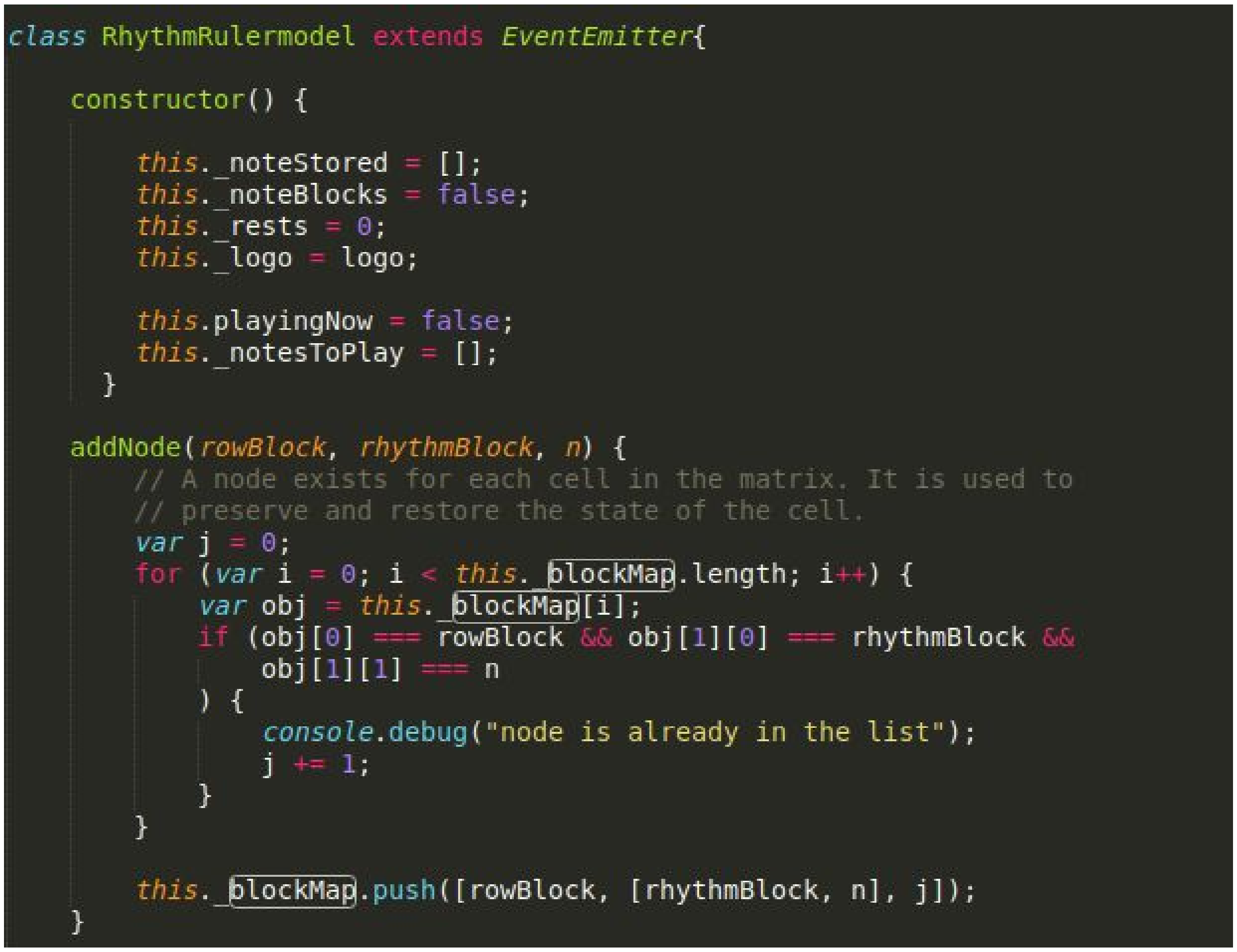
```
flow(args, logo, turtle, blk) {
    if (logo.temperament == null) {
        logo.temperament = new TemperamentControllerWidget(new TemperamentModelWidget());
    }
```

Abstract and basic controller will be look like



```
class RhythmRulerController extends EventEmitter{

    constructor(RhythmRulermodel) {

        this._model = Rhythmrulermodel;

        var widgetWindow = window.widgetWindows.windowFor(this, "phrase maker");
        this.widgetWindow = widgetWindow;
        this._view = widgetWindow;

    }

}
```

Abstract and basic model will look like:



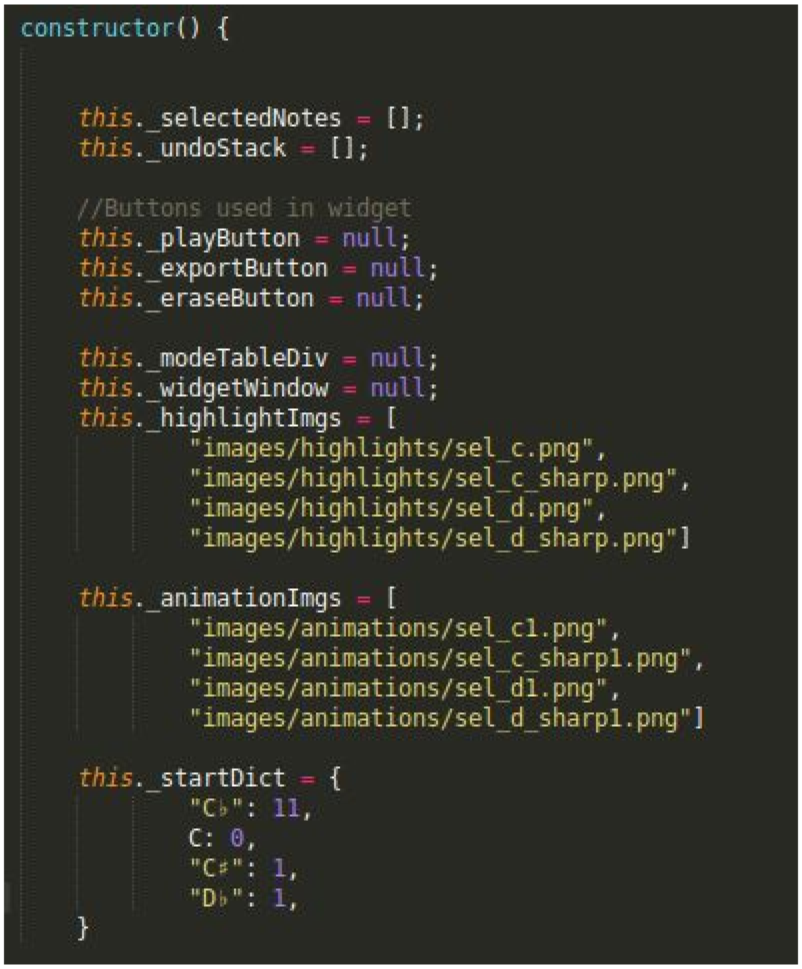
```
class RhythmRulermodel extends EventEmitter{

    constructor() {

        this._noteStored = [];
        this._noteBlocks = false;
        this._rests = 0;
        this._logo = logo;

        this.playingNow = false;
        this._notesToPlay = [];
    }

    addNode(rowBlock, rhythmBlock, n) {
        // A node exists for each cell in the matrix. It is used to
        // preserve and restore the state of the cell.
        var j = 0;
        for (var i = 0; i < this._blockMap.length; i++) {
            var obj = this._blockMap[i];
            if (obj[0] === rowBlock && obj[1][0] === rhythmBlock &&
                obj[1][1] === n
            ) {
                console.debug("node is already in the list");
                j += 1;
            }
        }

        this._blockMap.push([rowBlock, [rhythmBlock, n], j]);
    }
```

More methods will be added to the model of each widget. This is just an example with a limited variable for pitch time matrix widget. If some object which will store data needs to be added then it will be a matter of adding objects in constructor , adding related methods in model with proper modification .

Detailed Implementation and how it will work

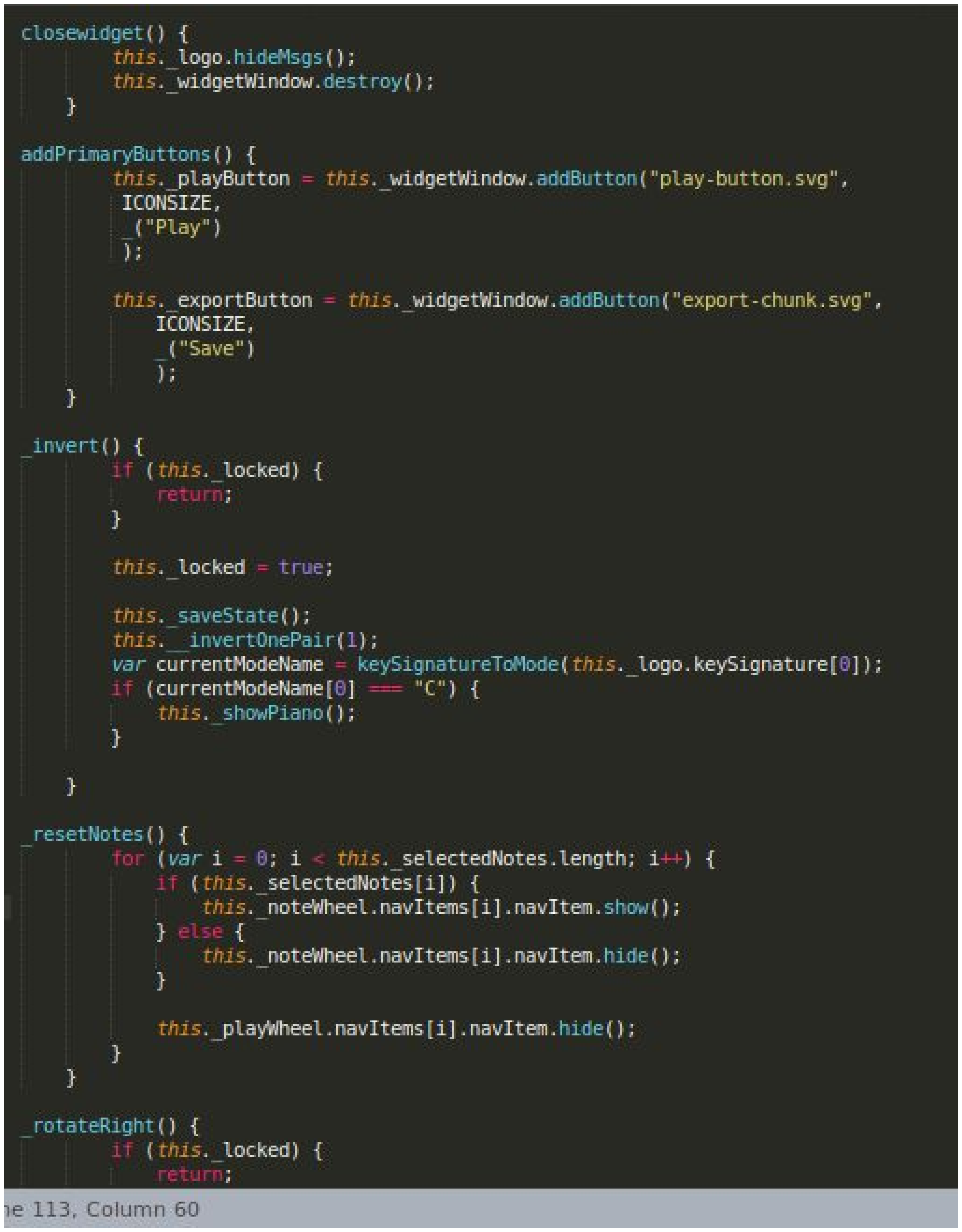
So far I have discussed what the model and controller will look like . Now I want to give details about how it is going to work . For each widget ,some methods are defined in their model.js file and the model cannot interact directly with view and vice-versa, but they will use controller.js file to perform tasks. So, controller.js file will have functions to associate with model.js and corresponding to the method called, it will also give commands to run view functions and change the output using view.js. Here , I am going to explain the working with the help of custom mode widget.

We start with a model class of custom mode widget.



```
constructor() {


    this._selectedNotes = [];
    this._undoStack = [];

    //Buttons used in widget
    this._playButton = null;
    this._exportButton = null;
    this._eraseButton = null;

    this._modeTableDiv = null;
    this._widgetWindow = null;
    this._highlightImgs = [
            "images/highlights/sel_c.png",
            "images/highlights/sel_c_sharp.png",
            "images/highlights/sel_d.png",
            "images/highlights/sel_d_sharp.png"]

    this._animationImgs = [
            "images/animations/sel_c1.png",
            "images/animations/sel_c_sharp1.png",
            "images/animations/sel_d1.png",
            "images/animations/sel_d_sharp1.png"]

    this._startDict = {
            "C♭": 11,
            C: 0,
            "C♯": 1,
            "D♭": 1,
    }
```

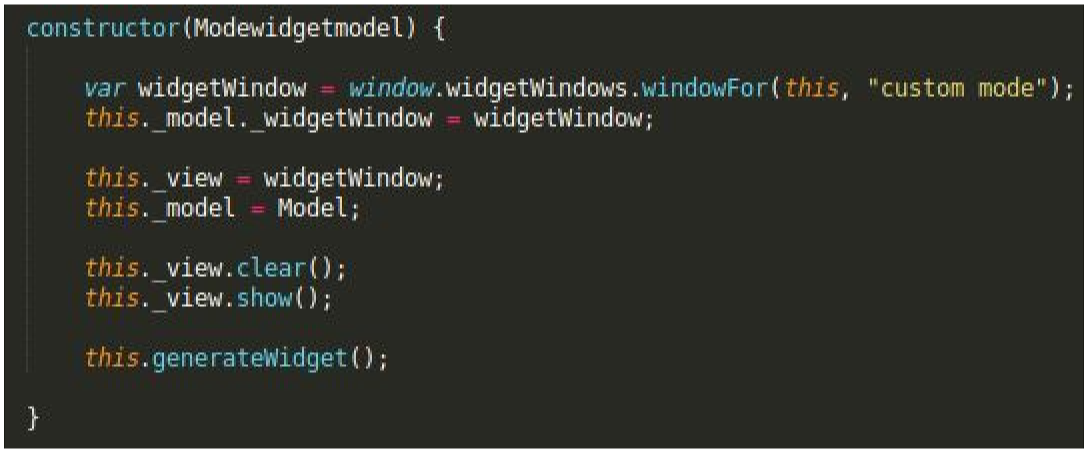
Model class of every widget will contain also various methods which will be called from the controller according to the user's input. In custom mode widget, there are many methods like _invert , _invert pair , _showPiano , _resetnotes, closewidget **,** etc . These methods can be called directly from the controller using this._model._methodname(); or can be called by other functions of model.js file, like this this._methodname(); In next page , I have added a screenshot of how these methods will be implemented.

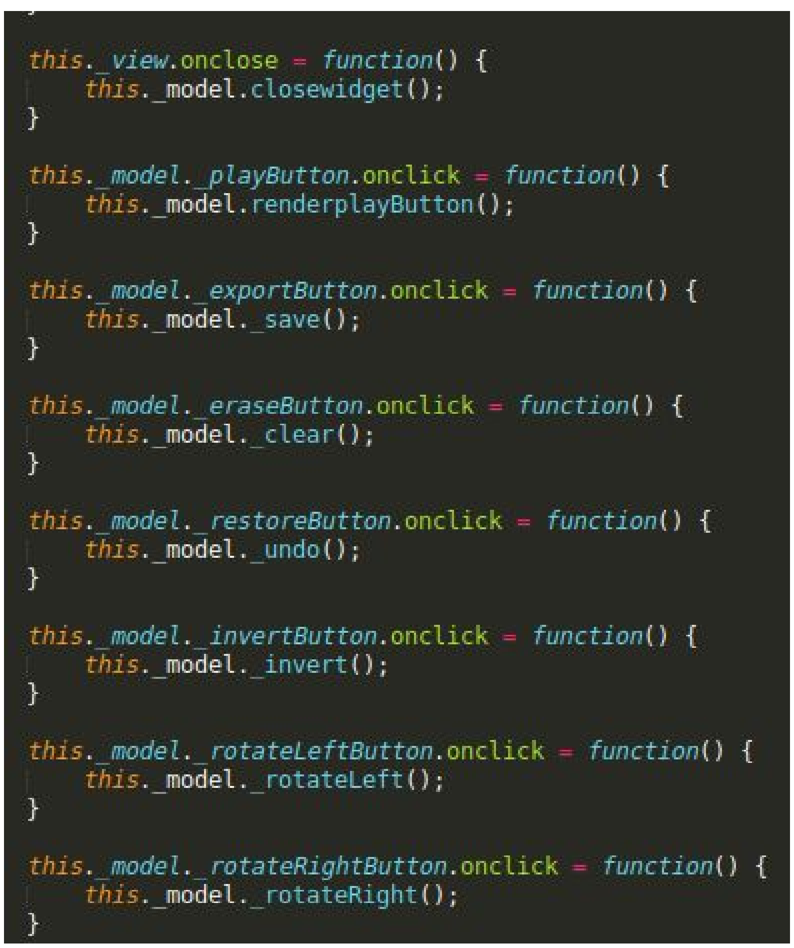


```
closewidget() {
        this._logo.hideMsgs();
        this._widgetWindow.destroy();
    }

addPrimaryButtons() {
        this._playButton = this._widgetWindow.addButton("play-button.svg",
         ICONSIZE,
         _("Play")
         );

        this._exportButton = this._widgetWindow.addButton("export-chunk.svg",
            ICONSIZE,
            _("Save")
            );
    }

_invert() {
        if (this._locked) {
            return;
        }

        this._locked = true;

        this._saveState();
        this.__invertOnePair(1);
        var currentModeName = keySignatureToMode(this._logo.keySignature[0]);
        if (currentModeName[0] === "C") {
            this._showPiano();
        }

    }

_resetNotes() {
        for (var i = 0; i < this._selectedNotes.length; i++) {
            if (this._selectedNotes[i]) {
                this._noteWheel.navItems[i].navItem.show();
            } else {
                this._noteWheel.navItems[i].navItem.hide();
            }

            this._playWheel.navItems[i].navItem.hide();
        }
    }

_rotateRight() {
        if (this._locked) {
            return;
```

As I said above methods will be called from controller.js , so let's talk about them. Controller will be responsible for creating relationships between model and view.

- **constructor** function which will create a model and view object by calling them ,it will also generate a basic widget window. Please see the first fig. of the next page.
- **methods** which will look for user interaction like onclick, onmousemove, onmouseout, onmouseover ,etc and call the corresponding methods in models to modify data and consecutively the view.Please see second fig. of next page.


```
constructor(Modewidgetmodel) {

    var widgetWindow = window.widgetWindows.windowFor(this, "custom mode");
    this._model._widgetWindow = widgetWindow;

    this._view = widgetWindow;
    this._model = Model;

    this._view.clear();
    this._view.show();

    this.generateWidget();

}
```


```
this._view.onclose = function() {
    this._model.closewidget();
}

this._model._playButton.onclick = function() {
    this._model.renderplayButton();
}

this._model._exportButton.onclick = function() {
    this._model._save();
}

this._model._eraseButton.onclick = function() {
    this._model._clear();
}

this._model._restoreButton.onclick = function() {
    this._model._undo();
}

this._model._invertButton.onclick = function() {
    this._model._invert();
}

this._model._rotateLeftButton.onclick = function() {
    this._model._rotateLeft();
}

this._model._rotateRightButton.onclick = function() {
    this._model._rotateRight();
}
```

I have also tried to refactor custommode widget along lines of M-V-C . Please see the link below.

Custom Mode widget Refactoring

Implementation details for other files (activity.js, blocks.js, block.js, palette.js,logo.js etc)

The major functional code of MusicBlocks project lies in activity.js. Regarding blocks creation , working files like block.js, blocks.js are responsible .For their smooth working logo.js is responsible. Other main part includes palettes.js . In this part of my project , I will refactor these files mentioned above which are :

- activity.js
- block.js
- blocks.js
- logo.js
- palette.js

I will be separating these files into model , view and controller .

Activity.js - As the code in this file is responsible for loading a new session (palette, toolbar , widgets) , mouse control , saving of project , saving of data in widget and much more . It has become a mess which I will be refactoring on the lines of MVC. I will be separating into three files aka **activitymodel.js , activityview.js and activitycontroller.js.**

1. **activitymodel.js** - this part will contain all data . for e.g. mode(beginner or advanced) , messages of load animation, cell sizes ,keyboard shortcuts, arguments passed in widgets etc. This part will also contain a definition of functions which will modify stored data. for e.g. preparExport , loadData , currentProject , etc.
2. **acitivityview.js** - this part will be responsible for generating music blocks and planet view based on the input . It will generate palette , toolbar buttons , trash, blocks container, turtles based on modes of the project.
3. **activitycontroller.js** - this part will be collecting mouse control definitions , user interaction , mouse control functions(mousemove , mouseon, mouseout,etc). Drag handler for blocks, trash handler . This will link the event to modify the model data and change the view according to the model data.

Block.js - From creating a new block , stroing data in it, and loading it again in new session this file is responsible . I will be dividing it into two parts of MVC (model and view).Controller part will be mostly covered in **blocks.js** and some part will be in its model.

1. **blockmodel.js -** this part will contain block data. for e.g. names of widgets, collapsible blocks, piemenu data. The defintion of functions related to block data modification and corresponding changes in view are stored in it. for e.g. change in block connection data, block in trash or not , updating functions (updatecache, newArtwork data update, etc).
2. **blockview.js -** this part will modify block representation on the window . Essentially , it will contain function to create artwork , regenerate artwok, show function for showing blocks, etc.

Blocks.js - This will be divided into three parts of MVC.

1. **blocksmodel.js** - this will contain block data list. Every information related to data will be stored here. For e.g. active block, list of loaded blocks , blocks in trash ,blockscale , etc.
2. **blockscontroller.js -** this will contain block movement controls .
3. **blocksview.js -** this will contain functions to extract blocks , relative movement of blocks,default block generation , block generation from previous session, etc.

Palette.js - This will be divided into three parts of MVC in the same file.

1. **palettemodel.js -** all data like default palette, blocks , mouse positions , scroll, active palette, palette text etc, will be here and related functions of the same to modify it .
2. **paletteview.js -** Function such as popdown palette to show a selected palette , update palette when mousemove on it etc are conatined here.
3. **palettecontroller.js -** this will set up event handlers in palette menu, mouse control options, update palette according to modes, blocklist, plugins added, etc.

Logo.js and other small files will also need refactoring according to MVC . I will do their refactoring as well . I am in touch with my mentor and as time goes with his suggestions , I will be doing it for any other files he wanted.

Part 2

Solving different issues of Music Blocks.

I will be solving issues related to widgets, planet view and adding features in music blocks. I also noticed that after successful completion of part 1 i.e. refactoring code along lines of MVC , it will be somewhat easier to tackle issues. It is also suggested by respected mentors . I want to start with some of my pending PRs to resolve the issue and follow the reviews by mentors .

I will be also testing all functionalities of every widget after their refactoring. One of the issues I would like to solve is related to phrase maker . Repeat blocks cause some notes to skip . I will take a close look and will solve the issue.

I also plan to add new music instruments to MusicBlocks . I have seen someone adding a viola during GCI that is the motivation behind this. For this , I will take help from my mentor and will add some instruments.

I have come to know about Musescore (a program for sheet music) while solving an issue related to barlines addition at the end . Solving that issue , I feel motivated to solve file corrupt error. This bug does not prevent you from opening the file in Musescore , just ignore the error message and file will open . For this issue, I gained some knowledge and came to know that Musescore is an open source community and for fixing this, the solution is to add some voice and copy the corrupted measure and replace it with the new voice and again do the replacement , now delete the new voice added to solve this issue. This works well but If somehow we can figure out some change in our .xml which does not make our file corrupted will be much better.

I have listed in the timeline the issue related to concentric rhythm ruler. In this issue , we first need to make alignment and discuss the details of implementation of two pie menus at the same time. This can be solved by adding code for both in the same place with change in size .

How will it impact SugarLabs?

This project aims to refactor code of Music Blocks project which is an important part of SugarLabs. This project will make the codebase of music blocks easy to understand and new features can be implemented easily. This will ensure that beginner developers who are new to this codebase will not face much difficulty. I also want to document my work and how widgets work after this refactoring during my GSoC only .This will make things easier.

What technologies (programming languages, etc.) will you be using ?

I will be working in plain javascript . My first main focus will be on refactoring widgets directory along the lines of M-V-C. I have got suggestions from mentor and Bottersnike (GCI participant) about race conditions in module bundling using “requirejs”. As suggested , I need to make use of define() to prevent race conditions . I will also keep in mind if strange race conditions start to appear on page load (especially a fresh load without cache).

<u>Timeline</u>:

In the first phase, I will try to complete refactoring of most of the widgets, and after that I plan to test them if they are working fine . Also suggested by the mentor, some work needs to be done in activity.js , block.js and logo.js files , palette code . I will be covering this part in the second phase .After that I will be solving bugs , issues and functionalities. Initially I plan to cover 2-3 widgets in a week and later 4- 5 widgets in a week. I will be giving 1-2 days per issue to solve them .

Please see my timeline:

S.no.	Date	Work
1.	May 4 - May 14	**Community bonding period.** ● Already familair with the code base . ● Discuss implementation details with the mentor. ● Complete the work started on custom mode

		widget refactoring . ● Take suggestions from mentors .
2.	May 15 - May 22	● Start refactoring rhythm ruler widget, pitch time matrix widget .
3.	May 23 - May 31	● Discuss with mentors about any ambiguity. ● Solve issue saving for widget for ruler widget and phrase maker.[#2017] ● Testing of widgets I have refactored.
2.	June 1 - June 7	● Refactor music keyboard, status widget. ● Discuss with mentor about any ambiguity.
5.	June 8 - June 16	● Cannot commit due to university exams. ● Active on IRC and mailing list. ● Solve issue saving in widget.
3.	June 17 - June 23	● Refactor meter, tempo and timbre widget. ● Also test working of functionalities.
4.	June 24 - July 3	● Testing of widget I have refactored. ● Refactor pitch slider, pitch staircase, temperament widget. ● Complete refactoring of all widgets.
5.	June 29 - July 3	**FIRST EVALUATIONS**
6.	July 4 - July 11	● Will refactor activity.js separate view part first. ● Discuss with the mentor about implementation.
7.	July 12 - July 19	● continue refactor blocks.js. ● Completely separate model and controller of it. ● Start working on logo.js .
8.	July 20 - July 26	● .I will focus on block.js and complete refactoring of logo.js ● Try to refactor them completely this week. ● Part 1 finished.
10.	July 27 - July 31	**SECOND EVALUATIONS**
11.	July 31 - August 6	● Will be completing any remaining work according to timeline provided. ● If work is already done , I will continue with the issues.
12.	August 7 - August 14	● solve issue [#1891] add a block note counter.

		● solve issue [#1875] clicks during musical keyboard based on 'meter' and 'tempo'. ● Above issue may take more time, understanding the concept of BPM . ● solve issue [#1290] circular rhythm ruler concentric when aligned. ● solve issue [#2051] Repeats skipped in phrase maker. ● solve issue [#1982] musescore corrupt file error. ● solve issue [#2165] bug of phrase maker widget. ● solve issue [#2162] bug in set Drum ● solve issue [#2162] bottom part of widget not visible by adding auto scroll.
13.	August 15 - August 24	● solve issue [#1738] variable modes ● issue [#1691] add more musical instruments to music blocks. ● I plan to add my first instruments in 3-4 days. ● Will be adding 5 more music instruments. ● add feature [#1673] export mouse animation and music keyboard together. ● add feature [#1508] pitch shifter. ● Also solve issues opened during this period. Will be adding more to this list.
August 24 - August 31		**SUBMIT CODE & FINAL EVALUATIONS**

During summer, I have no other commitment other than this project so I can allocate full 40-45 hours a week.

How many hours will you spend each week on your project?

I have no other commitments other than this in this summer. I am super excited to work on this project during GSoC 2020. I will spend 40-45 hours in a week to reach my goals mentioned on my timeline .During university examinations, I will devote 20-24 hours . I am also ready to work in the time according to my mentor's feasibility.

How will you report progress between evaluations ?

I will be writing blogs, weekly reports on medium platforms .My profile link on medium.com/@b18050 . I have seen previous year GSoC blogs in which they mention what they did in the last week, what they will do in the upcoming week and what are the difficulties they face , how they managed and overcome the hurdles . I have gone through them . I will be also writing such blogs , this is the best way I think I can report progress between evaluations. I am also ready to share weekly reports on the wiki page of Sugarlabs as I get to know about previous year GSoC projects from there only. I will also post my summary of work done during GSoC after successful completion of the project.

What are your expectations from us during and after the successful completion of the program?

I take this as a good opportunity to learn new skills and make strong new bonds with the community. I hope to complete my project in time and will be contributing to sugarlabs after GSoC also. As I am looking forward to participating in GSoC next year as a participant or may be as a mentor in GCI , GSoC . I also want to add some new activities in SugarLabs , after the completion , I want to focus on python part of it .In starting ,I will port sugar activity written in python to sugarizer in javascript like I have done some porting on Bounce activity of sugar to javascript. See this link.

Discuss your post-GSoC plans. Will you continue contributing to Sugar Labs after GSoC ends?

I will be contributing to sugarlabs after GSoC also. I want to apply again as a participant or may be as a mentor next year in GCI , GSoC . I want to become a member of this community by active participation. I am looking forward to working with this community for years.

Have you applied before in GSoC ? If yes, which organsiation?

No, this is my first time participation in GSoC.

Are you giving proposals to any other organisations this year?

I am giving this single proposal to SugarLabs community only.