[Purr Data]

Google Summer of Code 2023

Autosave Feature

Name and Contact Information

Name: Prakhar Agarwal

Github username: Prakhar-Agarwal-byte

Gitlab: @prakhar

Email: prakharagarwal3031@gmail.com

Alternative Mail: prakhar.agarwal.mec20@itbhu.ac.in

Discord username: @Prakhar#9227

Project Size: 175 hours

University info

- **University Name:** Indian Institute of Technology (BHU) Varanasi
- **Major:** Mechanical Engineering
- **Current year and expected graduation date:** 3rd year (junior). Expected graduation will be around 30th November 2024

Degree: B.Tech. (Bachelor of Technology (4 years))

Website: prakharcodes.tech

LinkedIn: prakhar-agarwal-byte

Country: India

Time Zone: IST (UTC+05:30)

Table of Contents

Overview

Abstract

When working on a project in Purr Data, saving regularly is important to avoid losing any changes or progress. The auto-save feature provides an added layer of protection by automatically saving the project at set intervals, such as every 10 minutes. In case of a sudden power outage or system crash, the auto-save feature can help to recover the project files, reducing the risk of losing valuable work. This feature also helps to reduce the likelihood of accidentally closing the program or forgetting to save changes. Overall, the auto-save feature in Purr Data is a helpful tool for ensuring that project data is automatically saved and backed up at regular intervals, providing peace of mind and added protection against data loss.

Goals

Ensure reliability: The autosave feature's primary goal is to ensure Purr Data's reliability. It should save the patch frequently and automatically so users don't lose their work during a crash or power outage.

Minimal disruption: The autosave feature should save the patch in the background without interrupting the user's workflow. This means it should not cause noticeable delays or slowdowns when saving.

File management: The autosave feature should keep track of the saved versions of the patch and make it easy for users to access and restore a previous version of their work if necessary.

User-friendly: The autosave feature should be easy to use and understand, even for users unfamiliar with Purr Data or programming in general. This means it should have a clear interface and provide helpful feedback to users.

How it's done on other software

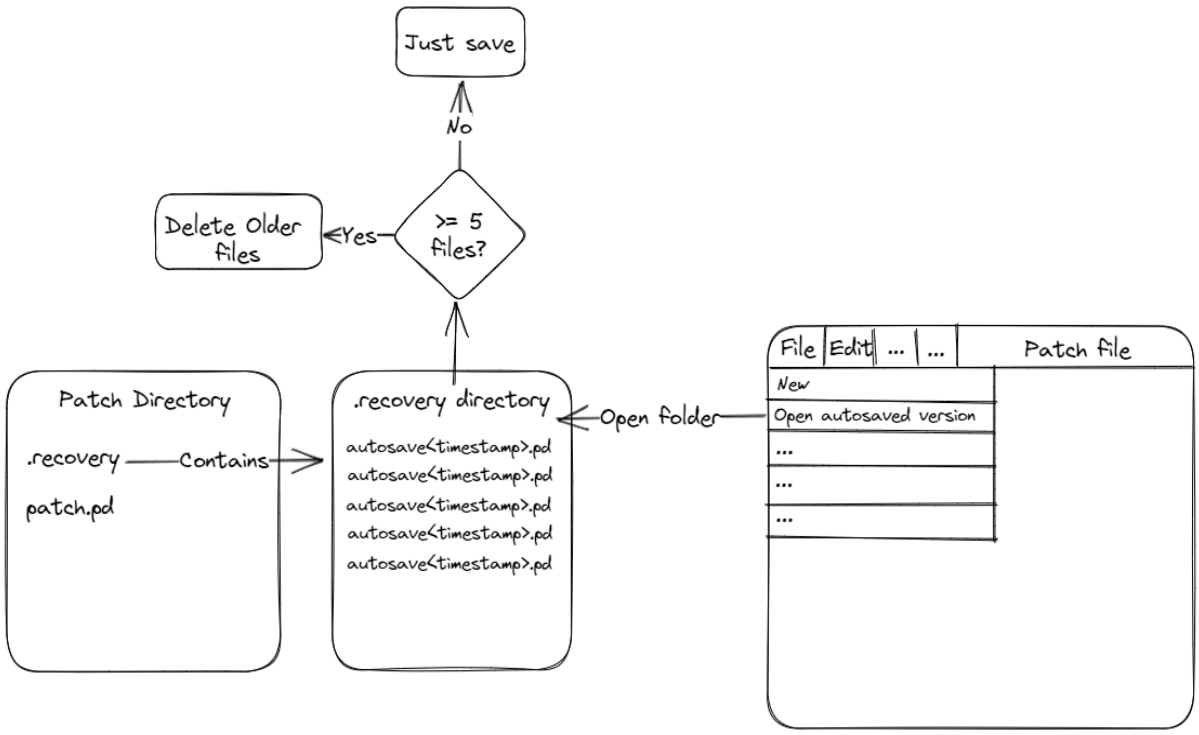
- **MS Word:** The files are saved after every 10 minutes at a predefined location in *APPDATA*. The user can select the files from the file menu he wants to recover. In case of a crash, he is also presented with a dialog to recover the latest files. The time and folder location for backups is customizable.
- **Sketch:** Sketch is a popular vector graphics editor, and digital design tool graphic designers and artists use to create high-fidelity digital designs. It creates a backup file with the extension ".sketchautosave" every 5 mins, which is customizable. These files are stored in a designated autosave folder. If Sketch crashes or closes unexpectedly, it will prompt the user with a dialog box to choose whether to open the autosaved version of your file or the last saved version.

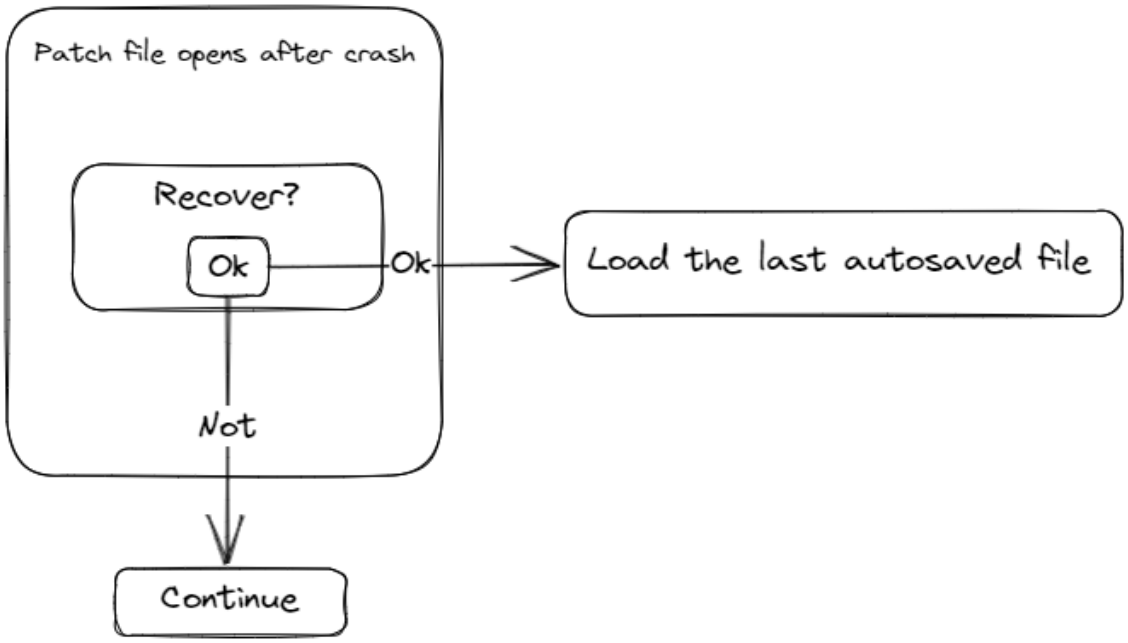
Approach

Implementation

I will store multiple patch versions in a hidden folder with timestamps. There will be a limit to the number of backup files stored, so they won't take up much space. I will add a menu option for the users to open the backup files. After clicking the menu option, a dialog will open pointing to the location of backup files regarding that particular patch. Now the user can select the specific backup file version he wants to recover.

Flowchart of the autosave feature:





Steps to implement the project plan:

- Create a timer in **pd_canvas.js** to save files every 10 mins.

```
// Call the saveToFileWithTimestamp function every 5 minutes
(300000 milliseconds)
setInterval(() => {
  saveToFileWithTimestamp('my_patch');
}, 600000);
```

- Write code to save files with timestamps to the preconfigured hidden folder.

```
function saveToFileWithTimestamp(filename) {
  // Create a timestamp string in the format YYYYMMDDHHmmss
  const timestamp = new Date().toISOString().replace(/[-:]/g,
'').slice(0, 14);

  // Construct the full filename with the timestamp
  const timestampedFilename = `${filename}_${timestamp}.pd`;

  // Call pdsend with the savetofile message and the timestamped
filename
  pdsend(`savetofile ${hiddenFolderPath}/${timestampedFilename}`);
}
```

If the backup files' number reaches the maximum limit, delete the oldest file first, then save the new one. Here the directory path is the path to the backup folder, and the prefix is the patch's name.

```
const fs = require('fs');

const directoryPath = '/path/to/directory'; // Replace with the
path to your directory
const prefix = 'filename-prefix'; // Replace with the prefix you
want to search for

fs.readdir(directoryPath, function(err, files) {
  if (err) {
    console.log('Error reading directory:', err);
    return;
  }

  const filteredFiles = files.filter(file =>
```

```
file.startsWith(prefix));

  if (filteredFiles.length === 0) {
    console.log('No files found starting with', prefix);
    return;
  }

  const oldestFile = filteredFiles.reduce((a, b) => {
    const statsA = fs.statSync(`${directoryPath}/${a}`);
    const statsB = fs.statSync(`${directoryPath}/${b}`);
    return statsA.mtimeMs < statsB.mtimeMs ? a : b;
  });

  console.log('Oldest file:', oldestFile);

  fs.unlink(`${directoryPath}/${oldestFile}`, function(err) {
    if (err) {
      console.log('Error deleting file:', err);
      return;
    }

    console.log('File deleted:', oldestFile);
  });
});
```

- Create a menu option in canvas to load backup files.

In pd_menus.js, append a new file menu item open backup files. Provide the label, key, modifiers, and tooltip

```
file_menu.append(m.file.open = new gui.MenuItem({
        label: l("menu.open"),
        key: shortcuts.menu.open.key,
        modifiers: shortcuts.menu.open.modifiers,
        tooltip: l("menu.open_tt")
    }));
```

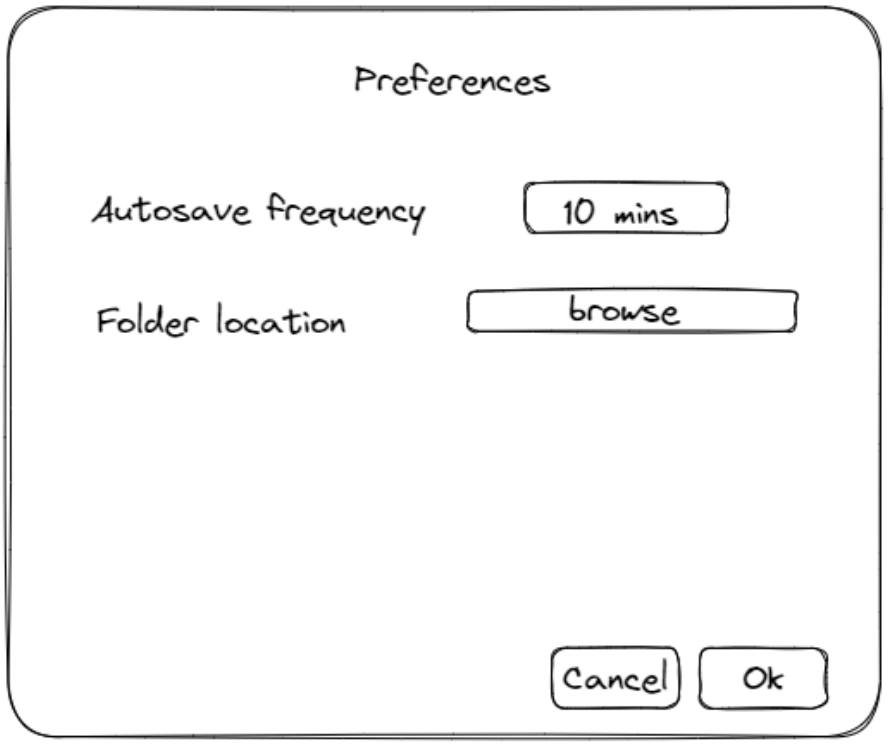
In pd_canvas.js, initialize the menu option to open up the backup file chooser dialog.

```
minit(m.file.open, {
        click: function (){
            var input, chooser,
                span =
w.document.querySelector("#fileDialogSpan");
            // Complicated workaround-- see comment in
build_file_dialog_string
            input = pdgui.build_file_dialog_string({
                style: "display: none;",
                type: "file",
                id: "fileDialog",
                nwworkingdir: "backup-folder-path",
                multiple: null,
                // These are copied from pd_filetypes in pdgui.js
                accept: ".pd,.pat,.mxt,.mxb,.help"
            });
            span.innerHTML = input;
            chooser = w.document.querySelector("#fileDialog");
            // Hack-- we have to set the event listener here
because we
            // changed out the innerHTML above
            chooser.onchange = function() {
                var file_array = chooser.value;
                // reset value so that we can open the same file
twice
                chooser.value = null;
                pdgui.menu_open(file_array);
                console.log("tried to open something");
            };
            chooser.click();
        }
    });
```

This will open a dialog pointing to the location of the saved backup files. Now the user can open up any backup file he wants to open.

- Create a window for customizing the autosave configuration

Develop a window using HTML, CSS, JS from where the user can change the his preferences for autosave such as the frequency of autosave and the backup folder location.



Project Timeline

Community Bonding and Onboarding(May 4, 2023 - May 28, 2023)	
`May 4, 2023 - May 28, 2023	• Discuss my project with my mentors and the community to see if any more improvements can be made in the plan of action. • Study how other softwares are implementing the autosave feature • I will start a blog documenting my journey with Purr Data. • Learn any extra tools and technology required for the project.
Project Period(May 29, 2023 - August 29, 2023)	
`May 29, 2023 - Jun 11, 2023	• Create menu option for autosave
Jun 12, 2023 - Jun 25, 2023	• Write code to create a timer to call save function after specified time

	• Code the function to save file in the backup folder location
Jun 26, 2023 - Jul 9, 2023	• Add support for deleting the backup files after the maximum limit is reached
Jul 10, 2023 - Jul 23, 2023	• Test if the files are getting saved properly in the right format • Load the saved files back and check if they are not corrupt or there isn't any data loss
Jul 24, 2023 - Aug 6, 2023	• Write labels, add shortcuts, and tooltips for the autosave menu option • Design the UI for preferences for the autosave feature.
Aug 7, 2023 - Aug 20, 2023	• Develop the UI and make the autosave configurable • Setup default time and backup location after discussion

	• Discuss stretch goals.
Aug 21, 2023 - Aug 28, 2023	• Complete stretch goals. • Finish up the documentation works. • Complete any remaining work and prepare my final report.

Extended Goals

1. Autosave patch every time an object is created.
2. View the complete autosave backup files history with timestamps.
3. Maintain a complete track of the changes in the Pd patch, making your recovery persistent.
4. Keep on adding new features and fixing bugs in Purr Data.

Benefits to Community

This feature can significantly reduce the risk of data loss due to accidental file closure, crashes, or other interruptions by automatically saving the user's work at regular intervals. This will improve the overall user experience and productivity by allowing users to focus on their projects without worrying about saving their work manually. Additionally, this feature will enhance reliability and accessibility, making Purr Data more trustworthy and user-friendly for a broader range of users. Improved collaboration is another benefit, as this feature will allow users to work together on projects without fear of losing their work.

Why Me?

I'm passionate about open source, and when I tried Purr Data for the first time, I instantly fell in love with it. I know the required technologies, and as I shared some of my accomplishments above, I've hands-on working experience with all of those. My passion for open source and never give up attitude when it comes to software development sets me apart from others. I am excited about the opportunity to work on this project and will work as hard as I have to make this project a grand success.

About Me

I am **Prakhar Agarwal**, a junior-year student pursuing Mechanical Engineering at the Indian Institute of Technology (BHU) Varanasi. I am a Full-Stack developer proficient in **HTML, CSS, Javascript, Reactjs, Nodejs, Python, Flask, C/C++, and Github**. I love exploring new tech and have worked with various technologies and languages. I love contributing to the open-source community and building something other developers can use.

Google Summer of Code 2021 with Purr Data

I was selected for Google Summer of Code 2021 with Purr Data organization. During the project, I improved the overall usability of the Purr Data Web Application to make it more intuitive, improved its stability by fixing many bugs, and improved the UI to fit the web app

better. Technologies I used in this project - **Javascript, HTML, CSS, NW.js, C++, Bootstrap**.
You can find details of my work on the GSoC project at
https://github.com/Prakhar-Agarwal-byte/purr-data

Contributor at Amahi

I have contributed to the Amahi Android app codebase and received a **recommendation letter from its co-founder, Carlos Puchol**. I worked on improving the UI/UX of the android app, fixed bugs, and enhanced many features in it. Technologies I used in this project - **Java, Kotlin, Android**

Contributions to other open-source projects

I've contributed to the **Meshery** and **Layer-5**, a cloud-native management plane offering lifecycle, configuration, and performance management of service meshes. I've also contributed to crypto wallets such as **Padawan Wallet,** where I learned more about blockchain and bitcoins. Here I **fixed issues, reviewed PRs,** and learned to **work collaboratively**.

Internships at startups

I have **done many internships at startups as a Full-Stack developer** where I got to work in a team and enhance my technical expertise with the help of my mentors. Technologies I used: **Javascript, Typescript, Reactjs, Nodejs, Python, Flask, Github, Expressjs, HTML and CSS**

List of my open-source contributions:

Organization	Contributions
Amahi	• PR • Issues
Purr Data	• PR • Issues
Padawan Wallet	• PR • Issues
Meshery	• PR • Issues
Layer-5	• PR