GSoC'25

Expanding Director Engine compatibility in ScummVM:

Director 4.0, Saving Director Files and Support for Director Games

Malhar Badve

About Me

Personal Info

Name	Malhar Badve
Email	malharbadve2046@gmail.com
Github	Malharbdv
Discord Name	SuperLemon
Mentor	Eugene Sandulenko (sev)
Location	India
Timezone	IST (UTC +05:30)

Educational Info

University	COEP Technological University
Major	Computer Engineering
Current Year	Junior (3rd)
Degree	Bachelor of Technology (B.Tech)

Hello! I am Malhar. I am a Third Year student (Junior) studying Computer Engineering at COEP Technological University, Pune. I've owned a computer all my life, I was always fascinated by programming, that's why I chose Computer Engineering as my major.

I'm very interested in games, game development and game engines. I've been part of some game development competitions. I was also the event head of the game development competition held at COEP.
I have studied Object-Oriented Programming, Computer Architecture, Operating Systems as course subjects. I am particularly interested in Operating Systems. I love the idea of Open Source Software. I find the idea very admirable and wish to be a part of it.

I have an experience in programming in the languages C, C++, Python, C# and JavaScript. I'm also familiar with git, make as well as cmake.

Availability

I'll be available anytime between:

12 AM IST (6:30 PM UTC) to 4 AM IST (10:30 PM UTC)
11 AM IST (5:30 AM UTC) to 12 AM IST (6:30 PM UTC)

Although I don't have any commitments (no internships or part time jobs) for the larger part of the GSoC project timeline. However, my 7th semester of computer engineering begins on 16th July, which will reduce my free time by a few hours for the later half of GSoC. We don't have the exact dates for when the exams will be held, it is safe to presume that they won't be until September.

Projects

Even though I've never worked on a project as big as ScummVM, I've made a bunch of small projects that may showcase my proficiency in c++ and object oriented principles. One of the reasons why I wish to contribute to ScummVM is that I feel like I will learn a lot about maintaining big projects, working with a team of more than 100 people and one that evolves constantly.

Here is my github profile which has some of the projects I have made in different domains like DSA, Computer Networks, Operating Systems, etc.
https://github.com/Malharbdv

Below are the projects I have made which are relevant to my proposal

This is a project that I did as part of my Data Structures course. It is a small tool to compress ASCII text files using Huffman Coding.
https://github.com/Malharbdv/dsa_compression_tool_cpp/tree/main

This is a small game that I made using Unity engine in C#. This is my introduction to Object Oriented Programming.
https://github.com/Malharbdv/Space_Shooter_Game

Pull Request

As per the requirement by ScummVM, I've made two Pull Requests.

1. I fixed a bug in the MPEG Decoder that was not correctly differentiating between PS (Program Streams) and ES (Elementary Streams). This was causing the initial logo video to not be played for the game Marvelous Mice Adventures: Meeting Sea Rat. I fixed it and it now works perfectly.

   Here is the link to the Pull Request on GitHub (merged):
   https://github.com/scummvm/scummvm/pull/6496

2. I have also been working on a director engine related intake task. Particularly the Panorama QTVR decoder, implementing three previously unimplemented features according to the original QTVR documentation for director engine, namely,
   - Changing Quality Mode: four different levels of quality modes including a dynamic quality mode which changes quality based on whether the user is interacting with the panorama.
   - Changing Warp Mode: no warping correction and one-dimensional warping correction for the cylindrical panorama.

   - Swing Transition: a swing animation for transition between two nodes with different values for FOV, tilt angle and pan angle.

   Here is the link to the Pull Request on GitHub:
   https://github.com/scummvm/scummvm/pull/6548

What attracted me towards ScummVM

I have been part of the Linux gaming community for some time. The name of ScummVM pops up often in that community. I like the fact the project helps towards preserving older Adventure games.

When I started contributing to ScummVM, I found that The ScummVM project seems to have a diverse set of tasks. There are Audio/Video Encoders/Decoders, Game Detection, Graphics System, Inputs, Accessibility like TTS, etc.

All of this is extremely organized, implemented cleverly and written well. The mentors I talked to were also very supportive. They tried their very best to solve all my queries even though all of them are working on ScummVM part-time.

What attracted me towards Director engine

The intake task I was assigned by sev was related to the Director engine. That's what made me look through the code for the Director engine. sev also suggested that I choose the Director engine for my GSoC proposal.

The Macromedia Director (originally named "VideoWorks") was dominant during the 90s and 2000s. This engine potentially targets a very large amount of games. It is still loved in the retro gaming community. It has its own fandom page.

The Director engine has been worked upon by a number of developers in the ScummVM community. Director Engine is also being developed very actively by the developers including rvanlaar, sev, moralrecordings and others. Reading their messages on the #engine-director channel gave me the impression that it will be fun to work on the Director engine and I'll get to learn a lot. team, and constant updates on the Discord channel convinced me that I will receive a lot of help from them when I start working on the Director engine.

The Project

Macromedia Director

Macromedia Director was a media application that allowed users to create interactive movies/games based on the movie metaphor.
It could incorporate many different bitmap, audio, and video file formats, making it possible to integrate media.
It used the Lingo scripting language to manipulate audio and sprites and other things like channels, frames, and castMembers.

This is a great start for learning how the Director engine works.

Various functionalities that were not built into Lingo could be used by adding

Xtras. An XDK (Xtra Development Kit) was provided to create C++ based Xtras, which could be used for File I/O, hardware calls, and advanced multimedia functionality.

The movies created could be exported to the .dcr, which could run on the shockwave plugin, or a binary executable could also be exported, which can
run naively on systems.

The two most important constructs of the Macromedia Director are the Cast and the Score. The Cast is a collection of CastMembers. Each CastMember can have
artwork like sprites, sounds, and colors contained in them which can be used
in the movie.

The Score consists of Channels and Frames. Channels are containers in the Score that can contain a CastMember, transition, or Lingo Command over the progression of time. Frames tell the state of the movie at a given time. The horizontal rows in a score are channels, and the vertical columns are frames. Their intersection creates cells.

Goal List

I will be completing the following tasks during the Google Summer of Code 2025:

- Workshop Movies : Completing support for all the workshop movies
- STUB Functions: Implementing all the stubbed functions
- Writing Director Files in ScummVM
- Director 4: Add support for SafeCracker and The Journeyman Project in the Director Engine
-

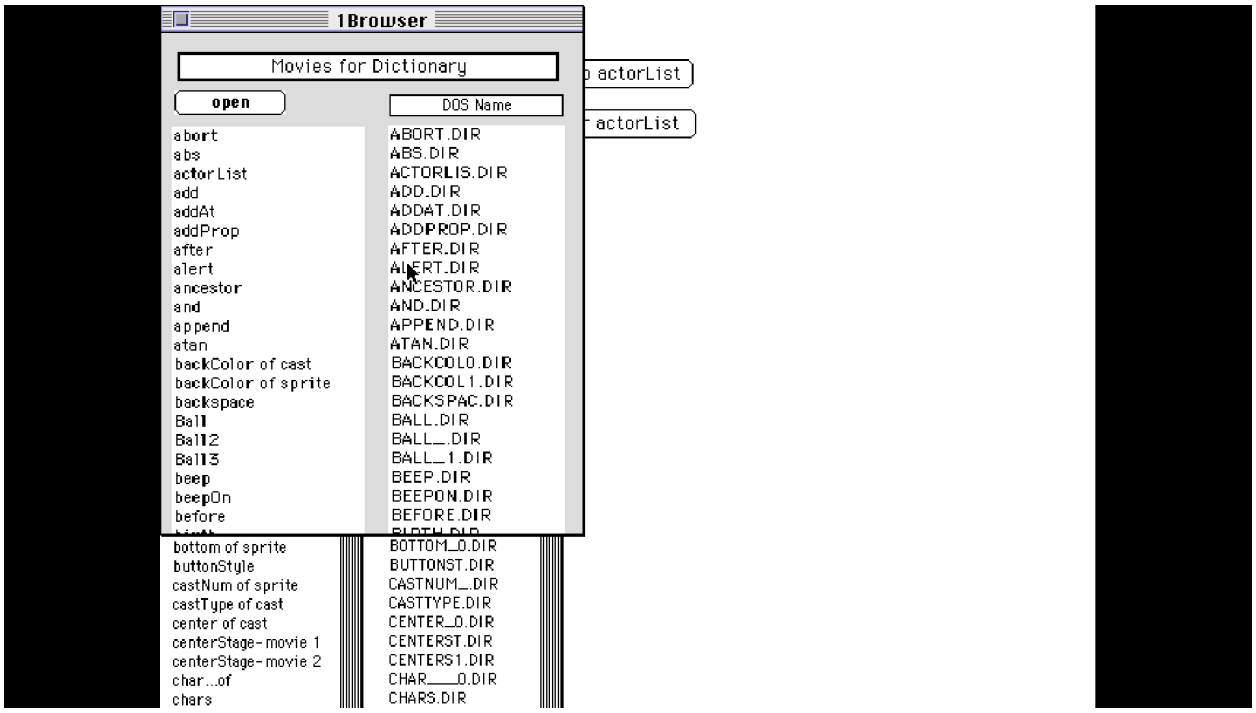
I would be focusing on support for Director 4 in the Director Engine and testing it out.
If time permits, I would also work on adding support for Director 5 movies by implementing functionality for the D5 workshop movies similar to that of D4. This would be my stretch goal. This document by djsrv is helpful for that.

1. Workshop Movies and Xtras

Workshop movies is a collection of 376 short movies to test the various functionalities of the Director Engine. Most of them are tests for some lingo commands. They are available for download at https://downloads.scummvm.org/frs/demos/director/

These workshop movies are available for both Director 4 and Director 5. A lot of them don't have their functionalities implemented. My first task would be to implement the functionalities of all the workshop movies.

As an example, the following screenshot shows the ScummVM window, when I open 0start.dir, which is supposed to let me open any of the 300+ short movies for demonstration, which unfortunately doesn't do its intended task.



This is a truncated list of all the short movies to test. I will go through each one by one and try to implement as many broken movies as possible during the self-assigned period as mentioned in the Timeline section below.

txt

```
-rwxrwxr-x 1  11K Jul  8  2020 abort.dir
-rwxrwxr-x 1 9.6K Jul  8  2020 abs.dir
-rwxrwxr-x 1  17K Jul  8  2020 actorlis.dir
-rwxrwxr-x 1  25K Jul  8  2020 addat.dir
-rwxrwxr-x 1  13K Jul  8  2020 add.dir
-rwxrwxr-x 1  13K Jul  8  2020 addprop.dir
-rwxrwxr-x 1  21K Jul  8  2020 after.dir
-rwxrwxr-x 1 3.2K Jul  8  2020 alert.dir
-rwxrwxr-x 1  16K Jul  8  2020 ancestor.dir
-rwxrwxr-x 1 2.7K Jul  8  2020 and.dir
-rwxrwxr-x 1  25K Jul  8  2020 append.dir
-rwxrwxr-x 1  25K Jul  8  2020 atan.dir
-rwxrwxr-x 1 7.2K Jul  8  2020 backcol1.dir
-rwxrwxr-x 1 7.4K Jul  8  2020 backcolo.dir
-rwxrwxr-x 1 2.8K Jul  8  2020 backspac.dir
-rwxrwxr-x 1 5.5K Jul  8  2020 ball_1.dir
-rwxrwxr-x 1 1.9K Jul  8  2020 ball.dir
-rwxrwxr-x 1 4.5K Jul  8  2020 ball_.dir
-rwxrwxr-x 1 8.8K Jul  8  2020 ball.mov
-rwxrwxr-x 1 3.6K Jul  8  2020 beep.dir
-rwxrwxr-x 1  12K Jul  8  2020 beepon.dir
-rwxrwxr-x 1  11K Jul  8  2020 before.dir
-rwxrwxr-x 1  14K Jul  8  2020 casttype.dir
-rwxrwxr-x 1  15K Jul  8  2020 center_o.dir
-rwxrwxr-x 1 9.3K Jul  8  2020 centers1.dir
-rwxrwxr-x 1  12K Jul  8  2020 centerst.dir
-rwxrwxr-x 1  20K Jul  8  2020 char___o.dir
-rwxrwxr-x 1  11K Jul  8  2020 chars.dir
-rwxrwxr-x 1  25K Jul  8  2020 chartonu.dir
-rwxrwxr-x 1  12K Jul  8  2020 checkbo1.dir
-rwxrwxr-x 1  12K Jul  8  2020 checkbox.dir
-rwxrwxr-x 1  16K Jul  8  2020 checkmar.dir
-rwxrwxr-x 1  12K Jul  8  2020 clearglo.dir
-rwxrwxr-x 1 9.6K Jul  8  2020 clickloc.dir
-rwxrwxr-x 1  11K Jul  8  2020 clickon.dir
-rwxrwxr-x 1  11K Jul  8  2020 close_wi.dir
-rwxrwxr-x 1  32K Jul  8  2020 colordep.dir
-rwxrwxr-x 1  11K Jul  8  2020 colorqd.dir
-rwxrwxr-x 1 2.5K Jul  8  2020 commandd.dir
-rwxrwxr-x 1  12K Jul  8  2020 constra1.dir
-rwxrwxr-x 1  10K Jul  8  2020 constrai.dir
-rwxrwxr-x 1 9.5K Jul  8  2020 contains.dir
.......... . .... ...  .  .... ............
```

2. STUBbed functions and XTRAs

There is a plethora of STUBs in the director engine code. Specifically the code for XObjects in xtras and xlibs.

These STUBs have to be implemented as these lingo commands are not being executed right now, due to which many director games and movies run with missing/broken functionality in ScummVM.

There are many cases of STUBed code. E.g.:

```
// engines/director/lingo/xtra/timextra.cpp
void Lingo::printArgs(const char *funcname, int nargs, const char *prefix) {
	Common::String s;
	if (prefix)
	s += Common::String(prefix);

	s += Common::String(funcname);
	s += '(';

	for (int i = 0; i < nargs; i++) {
	Datum d = _state->stack[_state->stack.size() - nargs + i];

	s += d.asString(true);

	if (i != nargs - 1)
	s += ", ";
	}

	s += ")";

	debug(3, "%s", s.c_str());
}

inline void printSTUBWithArglist(const char *funcname, int nargs) {
printArgs(funcname, nargs, "STUB: "); }

#define XOBJSTUB(methname,retval) \
	void methname(int nargs) { \
	g_lingo->printSTUBWithArglist(#methname, nargs); \
	g_lingo->dropStack(nargs); \
```

```
        g_lingo->push(Datum(retval)); \
        }

XOBJSTUB(MMovieXObj::m_copyFile, 0)

void TimextraXtra::m_new(int nargs) {
        g_lingo->printSTUBWithArglist("TimextraXtra::m_new", nargs);
        g_lingo->dropStack(nargs);
        g_lingo->push(g_lingo->_state->me);
}
```

Implementing the STUB code will also complete the Workshop Movies, as the non functional movies test STUB commands.
The Lingo documentation will be extremely helpful for this.

The STUBbed code also includes the code for a number of Xtras for Director engine like QTVR xtra and Timer xtra. In my second task, I implemented the code for the QTVR xtra by Apple, so I have an idea on how I can tackle this task.

3. Saving Director Movies in ScummVM

In the original Macromedia Director developers used to make content using Director and publish it on the Internet. Users could view the content using a browser plugin called Macromedia ShockWave Player. Hence, the Director movies (.dir files) can be converted to their compressed and protected versions (.dcr files) optimized for web delivery, requiring ShockWave plugin for playback.

In ScummVM's Director, we use a lot of the logic from this project by djsrv named [ProjectorRays](https://github.com/ProjectorRays/ProjectorRays) for the Director decompilation/deprotection of scripts. However, the saving of Director movies in ScummVM is missing.

The ProjectorRays project handles this saving function as follows:

- read a chunk off of the movie.
- keep the unprocessed data intact and output them verbatim.
- If the chunk is processed, find the chunk type (various types of chunks including Initial Map chunk, Config chunk, Cast info chunk, Cast member chunk, List chunk, etc.)

- If we know how to parse the part of the chunk, read them into a byte array as close to the disc implementation as possible, make the necessary changes and write them to disc again.

The relevent code in the github repo file.
e.g. The write function for the `ConfigChunk` class is showcased here.

This will need to be implemented in the ScummVM Director engine. Our current problem is that we do not keep things that we do not understand, like "unknown" fields, and even whole chunks in Director movies.

The complex part of this is that the ScummVM Director engine's data structures are not very close to the on-disk structures, this makes it harder for the Director Engine to save movies easily. i.e. the on-disk .dir and .dcr files are different from that loaded into Director. When loading a game/movie, ScummVM parses and converts the original binary data into its own in-memory representation, which may lose or change structure/format details from the original. One would need to reconstruct the exact original binary format, which the current structures don't support well.

This is the reading logic in the ProjectorRays project. This file contains all the logic of reading and writing a .dir file. This is close to the original data structure using RIFX container, storing data in typed chunks.
A `FOURCC` is a 32-bit identifier made from 4 ASCII characters* used to label chunks in binary files. In Director, FOURCC codes identify specific chunk types in `.dir`, `.dcr`, `.cst`, etc. This was followed very well in the ProjectorRays project.

This is the current loading logic of Director movies. This does not preserve the pre-existing structure of the .dir movie, instead tries to handle everything under one object.

On top of all the present chunk types in ProjectorRays, the Director engine might have to handle more chunk types according to original documentation.

I found some valuable documentation in Anthony Kleine's repository as well as the Earthquake Project repositories on which the ProjectorRays project is based on.
This is a good blog for introduction on the structure of .dir files.
This also gives a detailed guide on all the different chunks of a .dir file.

However, the ProjectorRays seems to be the best reference for accomplishing this task.

Currently the save functionality is stubbed in ScummVM Director engine lingo, it looks as follows:

```
// engines/director/lingo/lingo-builtins.cpp
void LB::b_save(int nargs) {
	g_lingo->printSTUBWithArglist("b_save", nargs);

	g_lingo->dropStack(nargs);
}

void LB::b_saveMovie(int nargs) {
	g_lingo->printSTUBWithArglist("b_saveMovie", nargs);

	g_lingo->dropStack(nargs);
}
```

4. Director Game Bugs

A large number of games were created using the Macromedia Director, ScummVM provides a list for the games detected correctly and can run to some extent.

I have noticed that two of the previous GSoC contributors have also worked on The Journeyman Project and Total Distortion, their blogs are here: hsk and raikou. Their blog posts outline a number of bug fixes in the Director engine to increase compatibility of these games. However, the games are still not "release-ready". I would like to continue these students' projects. Since there is already so much development done on these games, I believe that I would not be lost while working on these games.

I have access to the game demos for "The Journeyman Project" and "Total Distortion". I will watch gameplay walkthroughs for these original games and compare them to the current gameplay in ScummVM. Any functionality that seems missing must be added.

ScummVM's Director engine has limited support for The Journeyman

Project. I tested the Windows and Mac versions of the original game in ScummVM. In my few hours of gameplay in The JourneyMan Project demo, I have found the following bugs:

1. During the startup of the game, we see that a few xlibs have not yet been implemented

```
└─ ./scummvm
User picked target 'jman-demo-mac' (engine ID 'director', game ID 'jman')...
Running The Journeyman Project (CD Demo/Macintosh/English)
Journeyman Demo: 1ae45c23586b41997ba52e2e7c771c4c, 2880621 bytes.
WARNING: SearchSet::add: archive '/home/malhar/User_Drive/Projects/scummvm_gsoc2025_xtras/the_journeyman_project' already present!
Starting v311 Director game
WARNING: Window::probeResources: Couldn't find score with name: Michelani 645:Journeyman CD Demo:Demo 06!
WARNING: Lingo::openXLib: Unimplemented xlib: 'SetGlobalVolume'!
WARNING: Lingo::openXLib: Unimplemented xlib: 'XPlayPACo'!
WARNING: Cast::loadExternalSound: could not find external sound file rollingSnds!
STUB: Unhandled 'PICT' resource
WARNING: BUILDBOT: Uncaught Lingo error: Call to undefined handler 'XPlayPaco'. Dropping 2 stack items!
```

2. The demo has no sound. It could be using a particular unfinished xtra for sound generation.

Unfortunately I couldn't get my hands on an original copy of the The Journeyman Project game. However, this trello board also gives me a number of bugs that seem to be on the TODO list. Soon I will get access to these games and work towards fixing these bugs.

Working directly on a game to make it playable looks very interesting and complex. This directly impacts the support of Director games in ScummVM. Working on these will not only make the games themselves more playable but also solve a bunch of issues for any future games to be implemented in ScummVM. So, I would like to work on it.

Milestones

I can segregate my milestones into three parts like follows:

- Implementing STUB code and completing the Workshop Movies

  This is an early milestone, I will be focusing on getting as much STUBed code done as possible in the first few weeks, while simultaneously studying the other tasks. As the STUBed code may break functionality of a lot of director movies, I feel like adding the support for all the lingo commands in Director 4 will be very helpful in making the job of future game support easier.

- Completing the Saving functionality in Director Movies

  This will be my biggest milestone. The current director saving functionality is STUBbed. Refactor the director movie loading functionality to match the on-disk storage of .dir files as close as possible. Then figure out the writing of .dir files according to on-disk structure.

- Extending support for The Journeyman Project and Safecracker games

  The Journeyman Project and Safecracker games are still not fully supported and have bugs which are easy to encounter. They are also one of the prominent games made using Director 4. So making it largely playable will be a big milestone.

Deliverables

Following is the set of deliverables that I am planning on:

- Implementing the current STUB code so that existing Workshop movies can work as expected.

- Refactoring the Director movie loading code to match the on-disk structure as close as possible.

- Implementing the original FourCC system to store lingo compiled bytecode/other data into a .dir and .dcr files. Implement the chunk system of saving different information into a .dir file.

- Making the rest of the code compatible with the change in internal data structure of the Director movies.

- Make the Director engine in ScummVM capable of saving .dir/.dcr files with extensive testing.

- Fixing pre-listed bugs in The Journeyman Project and Safecracker games.

- Fixing other bugs which occur along the way in Safecracker and The Journeyman Project and make them largely playable.

GSoC Project Timeline

I want to go for the 350 hour format for this task.

I will try to stick to the following schedule to the best of my abilities. I understand that it is not realistic to map out the project down to the second.

I will consider the following timeline as a reference and a possible deadline for a given task.

The following timeline was made by analyzing the progress made by previous GSoC students and reading their blogs trying to find how much time they took for which task.

Date	Task
May 8, 2025 - June 1, 2025	Community Bonding. Discussion with sev and the director engine dev team about the project. With the Director engine code, start working on the STUB and workshop movies task.
June 2, 2025 - June 26, 2025	Complete as much of the STUB code and fix as many workshop movies as possible. Since these are small (but a large number of) tasks, I won't focus on it after the first few weeks. Prepare a plan of action for the Saving Director Movies task. Study the documentation.
June 27, 2025 - July 16, 2025	This is my project of focus. I will spend the most time on this task. Refactor the current loading logic of Director to include the different chunk types in director movies as different classes and their respective methods as described in ProjectorRays codebase.
July 9, 2025 - July 16, 2025	Midterm Evaluation. After this, my 7th semester begins.
July 17, 2025 - July 30, 2025	Implement the FOURCC chunk codes and accordingly segregate the information stored in a .dir file in ScummVM director.
July 31, 2025 - August 15, 2025	Implement the rest of the supporting functions in ProjectorRays codebase and port them to ScummVM Director engine. Modify other functionality that depends on the current data structure.
August 16, 2025 - August 30, 2025	Extensively test the newly implemented functionality of saving .dir files. Fix other bugs that may occur. Padding time to make sure it gets implemented completely.
August 31, 2025 - September 8, 2025	Begin work on The Journeyman Project and Safecracker games. Fix the incomplete functions while loading up The Journeyman Project game including the xlibs `setGlobalVolume()` and `XPlayPACo`

September 9, 2025 - October 2, 2025	Extended period for contribution begins. I want to focus on supporting Director 4 games as much as possible in this period. Work on the already reported bugs in the Trello board or elsewhere. Make the games as comparable to the original as possible.
October 3, 2025 - October 19, 2025	Add sounds to The Journeyman Project. Implement the corresponding xtras. Continue to fix bugs in the JourneyMan Project and Safecracker. Make them as close to being playable as possible.
October 20, 2025 - November 2, 2025	Off time, possibly exams in this period. If not, consider this period as a padding period in case some of the tasks above take more time than planned.
November 3, 2025 - November 10, 2025	Final Evaluation. Documentation and code finalization.

I have kept ample time to mitigate any unexpected blockers. I have also mentioned the schedule for my 7th semester at university. If everything goes better than expected, I would move onto my stretch goal - Adding support for Director 5 in ScummVM. I won't be completing this in the GSoC timeline, but would rather initiate work on it and continue developing along with the dev team Post-GSoC too. Director 5 support is in very initial stages in ScummVM.

Benefits to the ScummVM community

The Director Engine would be adding support for a lot of games to ScummVM, increasing the number of games that ScummVM can run. I'll be documenting every single piece of code that I have to deal with which I believe will help any upcoming

I am also interested in contributing to the Director Engine Post-GSoC at which point I would have a lot of knowledge of the codebase, to make it good enough to be released, fix bugs, or add functionality. I am also hoping to work on other components of ScummVM as a part of the dev team.

I feel like this will help a lot of upcoming development on the Director engine in ScummVM. Keeping the vastness and the sheer number of supported games in Director, the development of the Director engine will continue for years to come.

What makes me the best person to work on this project?

I am a persistent programmer. I don't like giving up things easily, if at all. I give my 100% to anything that I might be working on. I also like taking on complex tasks. This has instilled a confidence in me that as long as the task at hand is reasonably possible, I can do it.

I primarily code in C++ and Python and am pretty familiar with C++ internals and the Object-Oriented paradigm of C++. I am also familiar with build tools, Make, and gdb to aid me in programming and debugging.

I joined the ScummVM Discord server soon after the GSoC organizations were announced and have contributed to ScummVM since early March. Most of the time, I have been skimming the code of various components, understanding how engines work. My intake task was very closely related to the Director engine which gave me a good idea of how the Director engine works in ScummVM. I have already made some contributions to fixing existing bugs and adding functionality in the Director Engine. I have also gone over the ProjectorRays project decently well. I have an idea on how it works and how I will implement it in ScummVM.

I also enjoy reading and writing documentation which will help the community understand the Director engine better.