Ryan Hammett

Spring 2016 Senior Design Project

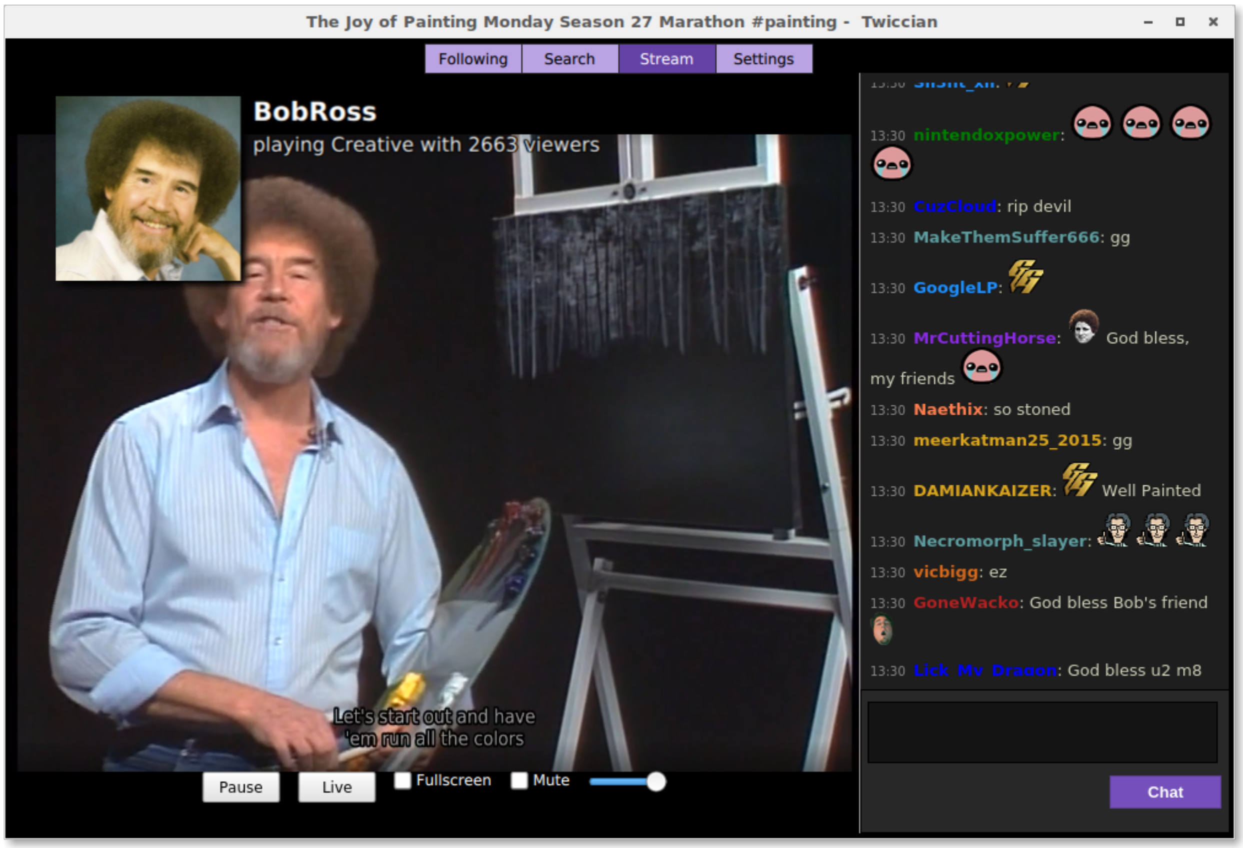
TWICCIAN

A TWITCH CLIENT FOR LINUX

Twiccian is a native app written for Linux to allow the user to watch Twitch.tv streams without the use of Adobe Flash or a web browser. Twitch is a very common platform for streaming games, speedruns and more recently general creativity, but suffers from the fact that the web player and chat are built on Flash. It is well known that this has an impact on the battery life, security, and system usage of many computers, and this is the problem Twiccian tackles.

Abstract

Twiccian is a native app written for Linux to allow the user to watch Twitch.tv streams without the use of Adobe Flash or a web browser. Twitch is a very common platform for streaming games, speedruns[1] and more recently general creativity, but suffers from the fact that the web player and chat are built on Flash. It is well known that this has an impact on the battery life, security, and system usage of many computers, and this is the problem Twiccian tackles.


The Joy of Painting Monday Season 27 Marathon #painting - Twiccian
Following
Search
Stream
Settings
BobRoss
playing Creative with 2663 viewers
Let's start out and have 'em run all the colors
Pause
Live
Fullscreen
Mute
Chat

13:30 nintendoxpower:
13:30 CuzCloud: rip devil
13:30 MakeThemSuffer666: gg
13:30 GoogleLP:
13:30 MrCuttingHorse: God bless, my friends
13:30 Naethix: so stoned
13:30 meerkatman25_2015: gg
13:30 DAMIANKAIZER: Well Painted
13:30 Necromorph_slayer:
13:30 vicbigg: ez
13:30 GoneWacko: God bless Bob's friend
13:30 Lick_My_Dragon: God bless u2 m8

Figure 1: Twiccian stream view with the Bob Ross channel

[1] Beating a game to credits or completion as fast as possible.

1. Introduction

This paper will cover the changes and improvements made on Twiccian. For more in-depth information on Twitch.tv and the creation of Twiccian, please view the original report.[2] For the Spring 2016 semester the focus for the development of Twiccian was to add more features. This was a daunting task, as the development team of Twiccian was cut from three members to one. This pushed back development until enough of the application was understood to make enhancements. Once research was completed, the first task was to allow the daemon to use IRCv3[3]. What IRCv3 allows access to is the new tagging features, these give users tags based on their account status. For example, if a user has a set color for their chat name it will hold a color tag [Figure 1]; if the user is a channel moderator they will have a user state tag [Figure 4]. These tags open up the possibilities of a chat more like the chat used on the Twitch website. Then, after enough of the daemon was refactored to allow for IRCv3 the focus was pushed to more visible features on the frontend. This included updated views for Following, Search, and Stream alongside several new API calls.

Twiccian has been released under version 1.1 for this semester and is available in the Arch User Repository[4]. Due to the packages required to build and run this application, Twiccan is still only available on Arch Linux. For future work on this application the daemon and frontend will need to be combined into a single application. The current structure of the applications limit the possibilities of future features to be on par with Twitch. This will also be the optimal time to switch programming languages,

[2] https://goo.gl/fVN7xR
[3] "IRCv3 Specifications"
[4] https://aur.archlinux.org/packages/twiccian/

but the GUI will still be developed in Qt. For now, Twiccian will be left as is until further collaboration can be determined.

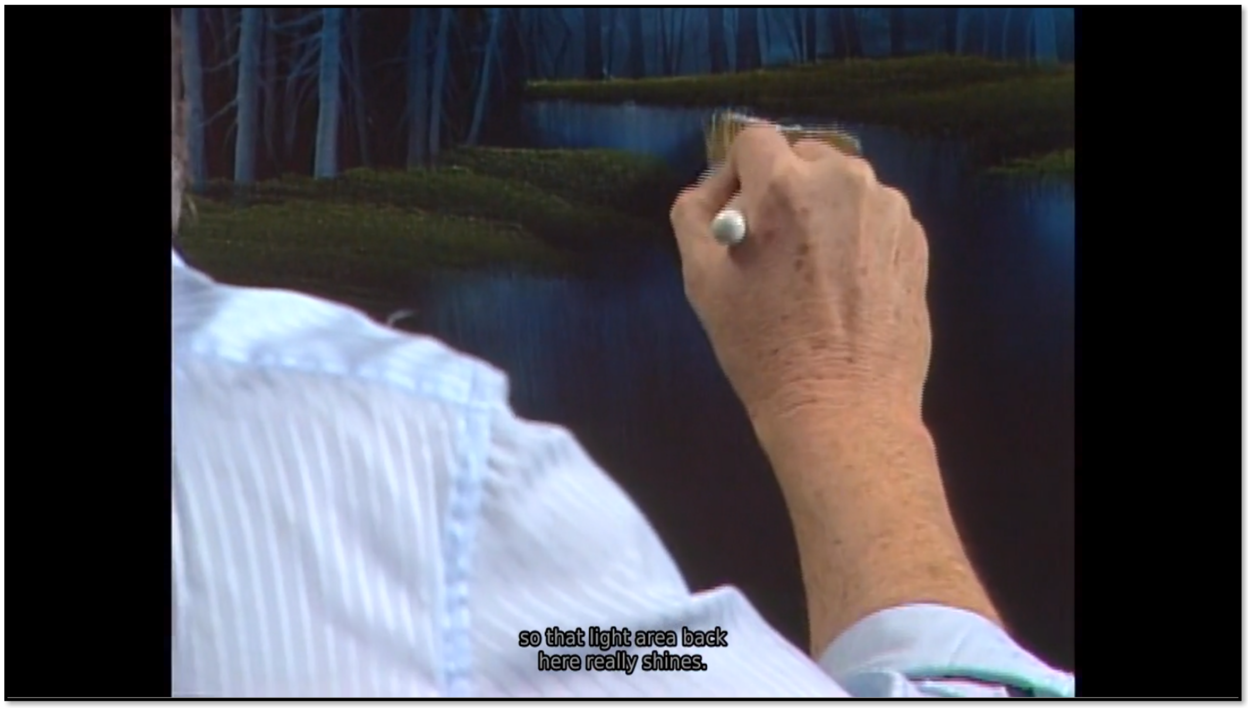


Figure 2: Full screen of Stream view

2. Background

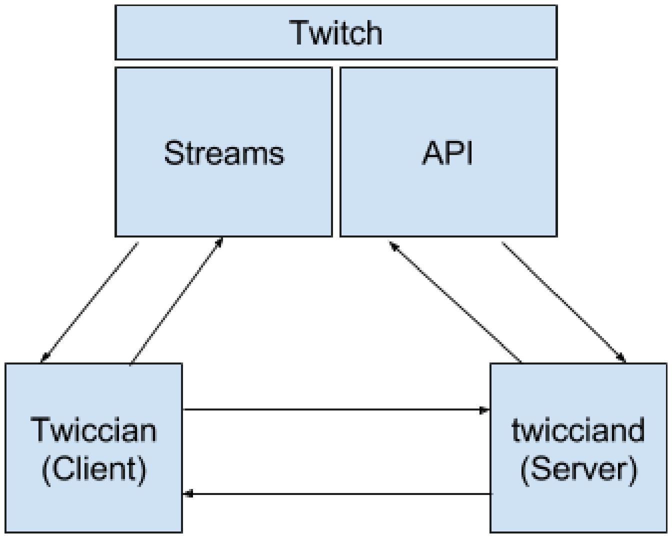
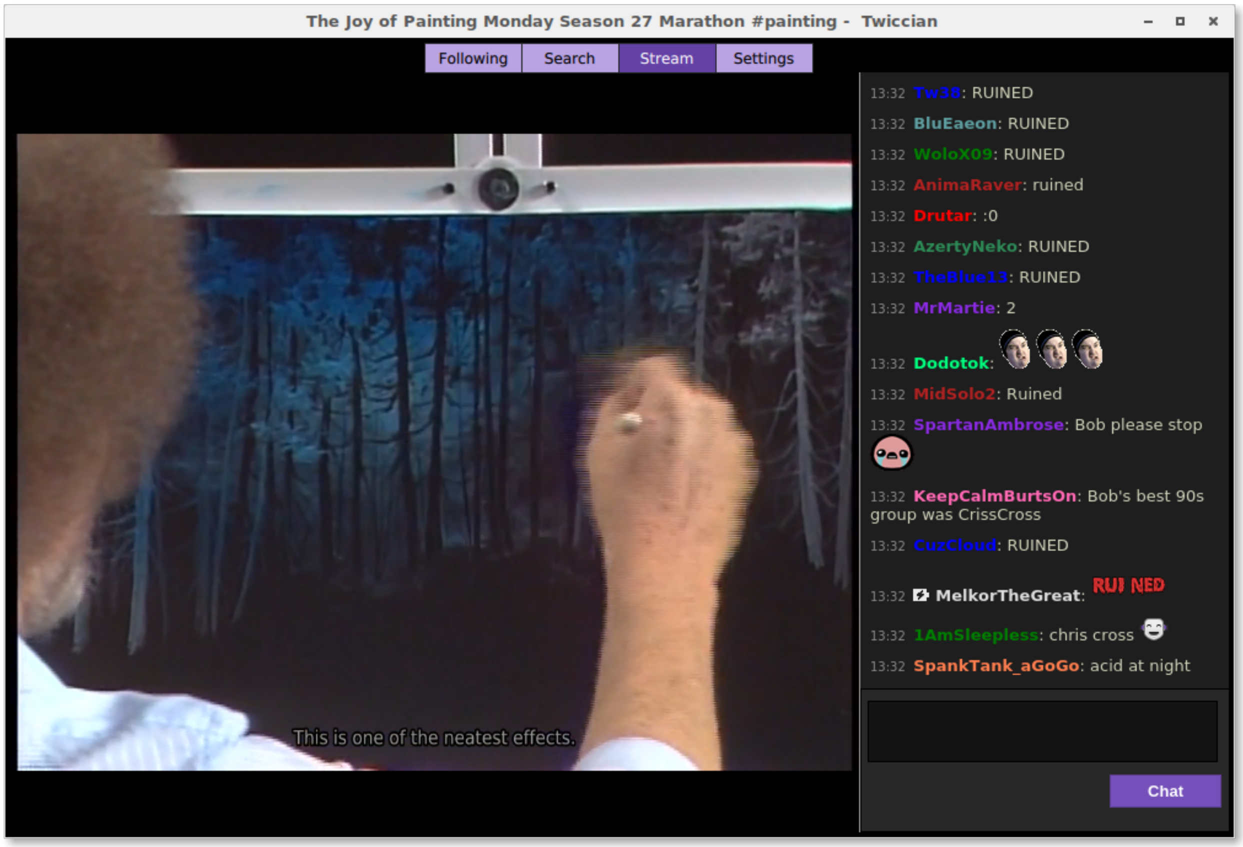


Figure 3: Diagram of Twiccian's architecture

Much of Twiccian and twicciand have not changed in terms of the applications framework and operation [Figure 3]. The structure and flow of communication and information passing have also largely stayed the same. The only changes to note deal with the updated chat in both the daemon and frontend. The sorcix/irc Go library was switched to the IRCv3 testing branch for access to the IRC commands in Go.[5] This allowed for the commands to be sent to the Twitch server to request access to the new tagging features. Once the new calls were permitted, new functions were implemented to parse these messages to be sent to the frontend to read from span data tags. The chat frontend needed to look for these data tags and also cache the global badges to visibly show these new features [Figure 4].


The Joy of Painting Monday Season 27 Marathon #painting - Twiccian
Following
Search
Stream
Settings
This is one of the neatest effects.
Chat

13:32 Tw38: RUINED
13:32 BluEaeon: RUINED
13:32 WoloX09: RUINED
13:32 AnimaRaver: ruined
13:32 Drutar: :0
13:32 AzertyNeko: RUINED
13:32 TheBlue13: RUINED
13:32 MrMartie: 2
13:32 Dodotok:
13:32 MidSolo2: Ruined
13:32 SpartanAmbrose: Bob please stop
13:32 KeepCalmBurtsOn: Bob's best 90s group was CrissCross
13:32 CuzCloud: RUINED
13:32 MelkorTheGreat: RUI NED
13:32 1AmSleepless: chris cross
13:32 SpankTank_aGoGo: acid at night

Figure 4: Chat user seen with a turbo badge

[5] Demuzere, Vic, Go irc package

3. Design & Implementation

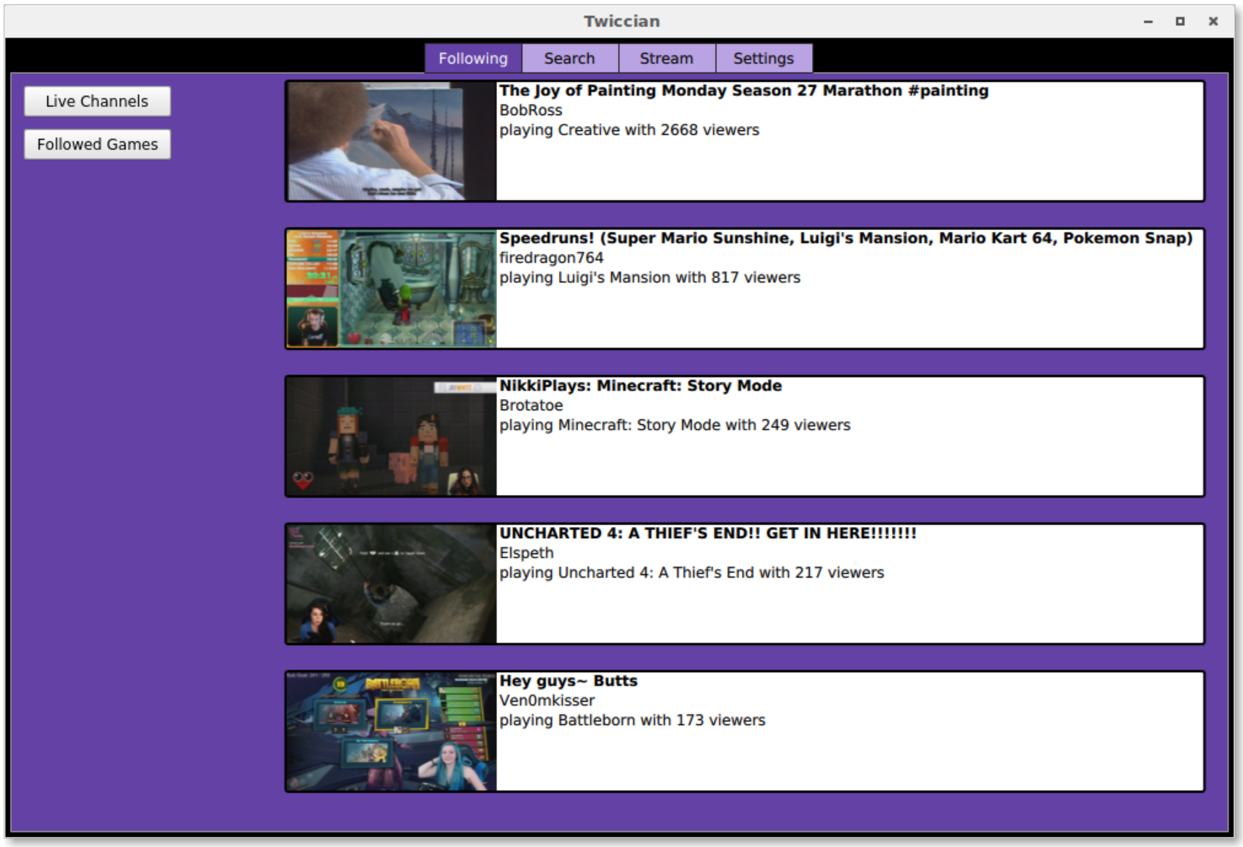
3.1. Organization

For this project several services were used in the development of this application. Git and GitHub housed the project, and the issue tracker was used to list bugs and problems with Twiccian and twicciand. An iOS app called 2Do was used to manage intended features and set due dates for the application. On several occasions, Slack was used to communicate with past members on certain topics and questions on this application.

3.2. Workspace

The computers used for this application ran Antergos with Cinnamon desktop. The languages used for Twiccian were QML, C++, HTML5, CSS, and JavaScript. The language used in twicciand was Go. To edit these languages, Vim and Qt Creator were used to write and edit the files for the applications in this project.

3.3. Application Features



The Joy of Painting Monday Season 27 Marathon #painting
BobRoss
playing Creative with 2668 viewers

Speedruns! (Super Mario Sunshine, Luigi's Mansion, Mario Kart 64, Pokemon Snap)
firedragon764
playing Luigi's Mansion with 817 viewers

NikkiPlays: Minecraft: Story Mode
Brotatoe
playing Minecraft: Story Mode with 249 viewers

UNCHARTED 4: A THIEF'S END!! GET IN HERE!!!!!!!
Elspeth
playing Uncharted 4: A Thief's End with 217 viewers

Hey guys~ Butts
Ven0mkisser
playing Battleborn with 173 viewers

Figure 5: Live Channels in Following view

The Live Channels take the updated data blocks to show the stream information in a more favorable view [Figure 5]. With Qt 5.6, the text rendering was updated to be more compact.[6] These data blocks are also used in the search results and prevent previous crashes by catching null games, views, or titles.

[6] "Qt QML 5.6"

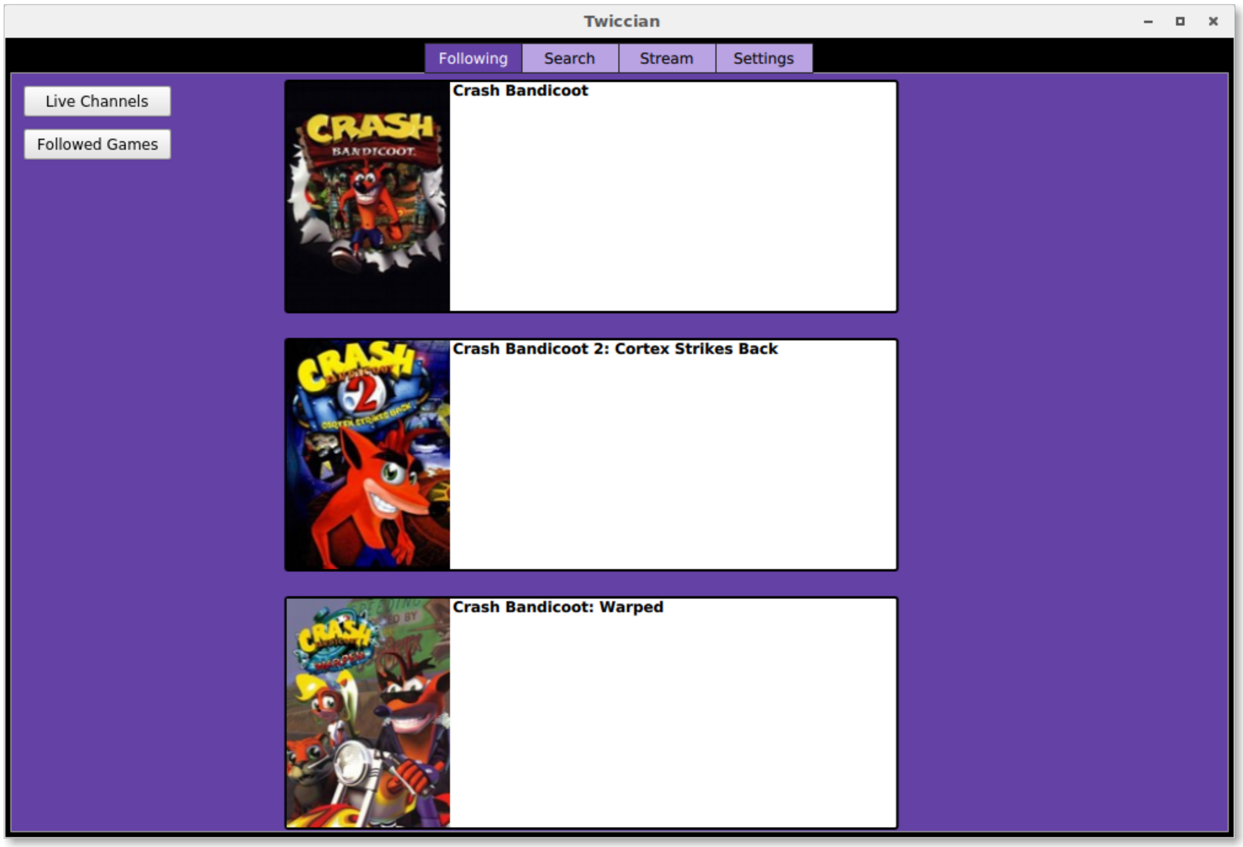
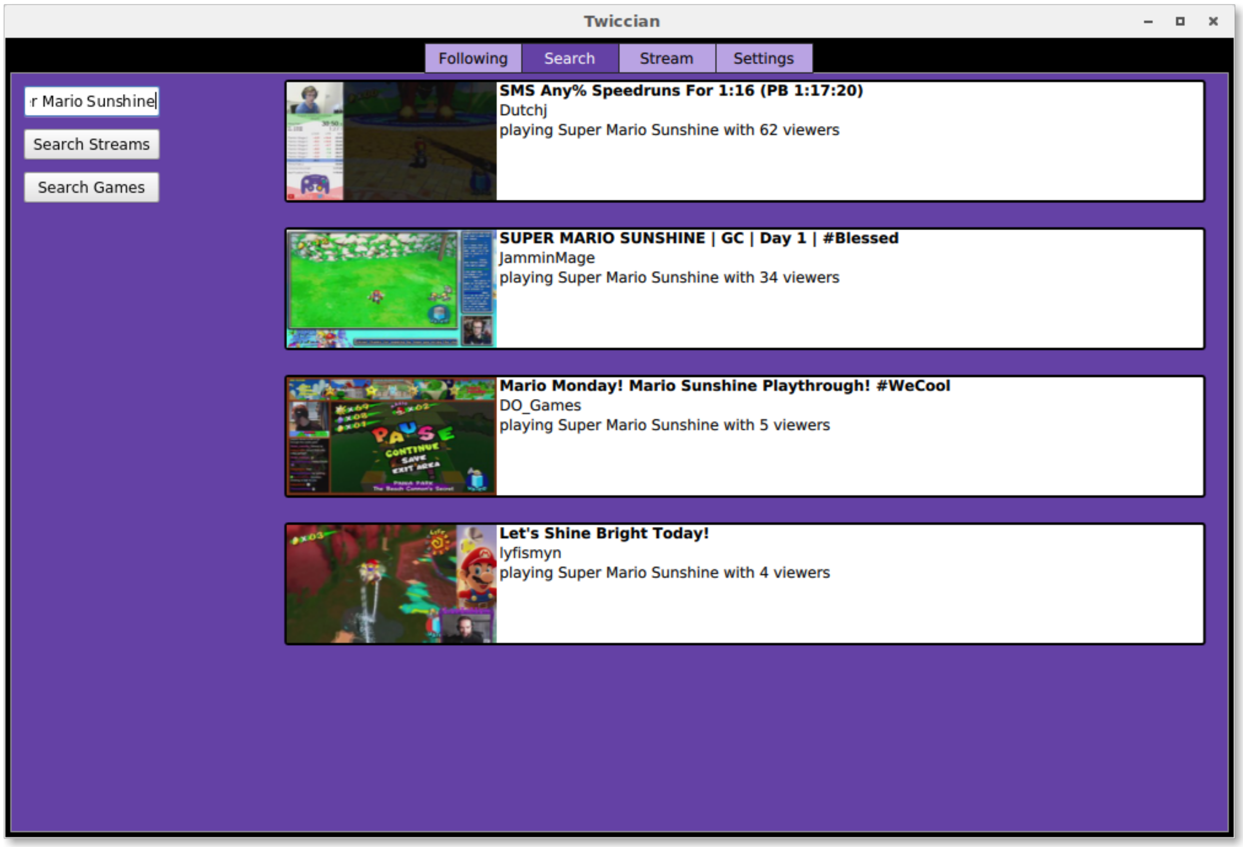


Figure 6: Followed Games in Following view

Followed Games is a new view added since last semester [Figure 6]. The API call for followed games is still in testing and not available on the API documentation.[7] The current structure of the website allows Twitch to show only games that contain live channels. For Twiccian to do the same, the daemon would need to be merged with the frontend to grab the search results asynchronously for live channels. These data blocks link to a search of the specified games, returning nothing for no live channels.

[7] "Justintv/Twitch-API"



SMS Any% Speedruns For 1:16 (PB 1:17:20)
Dutchj
playing Super Mario Sunshine with 62 viewers

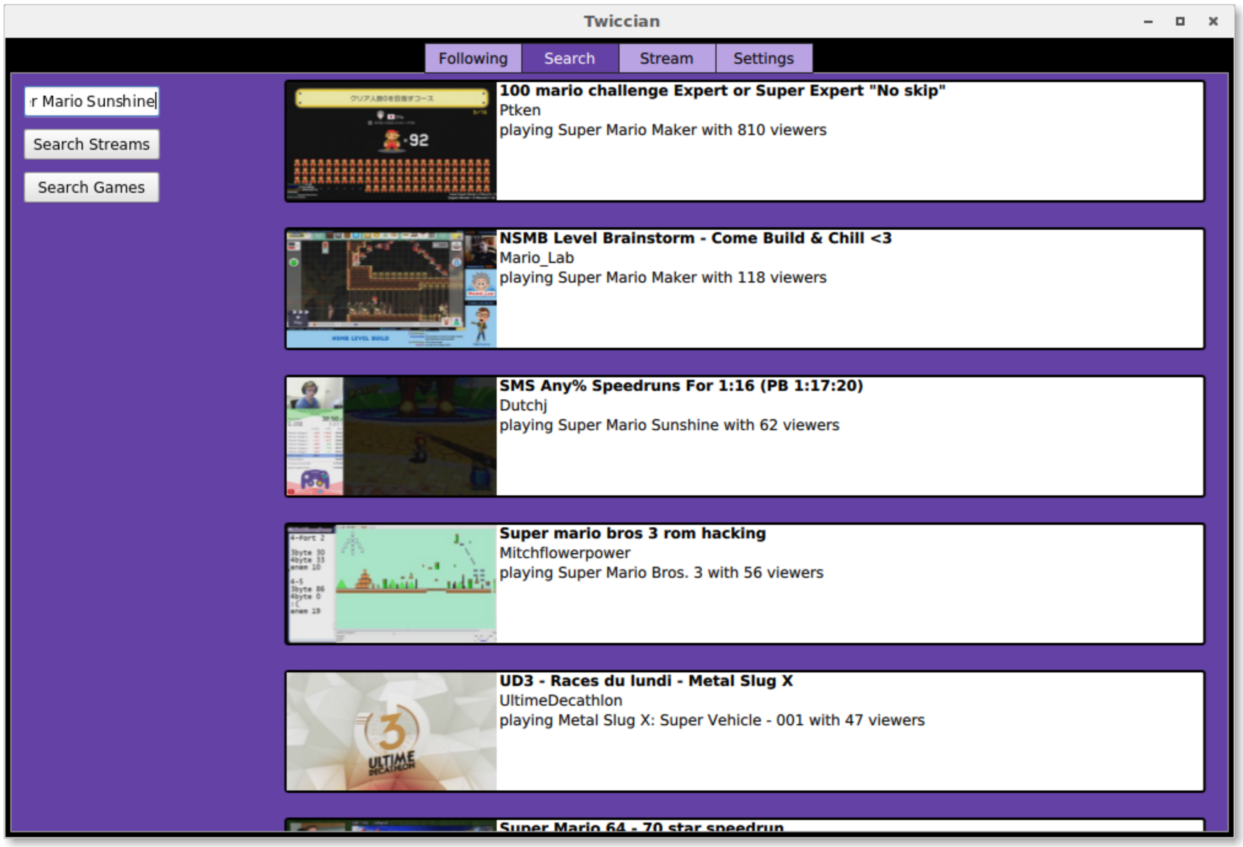
SUPER MARIO SUNSHINE | GC | Day 1 | #Blessed
JamminMage
playing Super Mario Sunshine with 34 viewers

Mario Monday! Mario Sunshine Playthrough! #WeCool
DO_Games
playing Super Mario Sunshine with 5 viewers

Let's Shine Bright Today!
lyfismyn
playing Super Mario Sunshine with 4 viewers

Figure 7: Results from a game search

Using the Search Games button or Followed Games view in Following will result in a search result of only the specified game in the query [Figure 7]. Unlike the Search Streams button, this will return only the game that is currently being specified by the channel. The information displayed is similar to that of the Following view [Figure 5].



100 mario challenge Expert or Super Expert "No skip"
Ptken
playing Super Mario Maker with 810 viewers

NSMB Level Brainstorm - Come Build & Chill <3
Mario_Lab
playing Super Mario Maker with 118 viewers

SMS Any% Speedruns For 1:16 (PB 1:17:20)
Dutchj
playing Super Mario Sunshine with 62 viewers

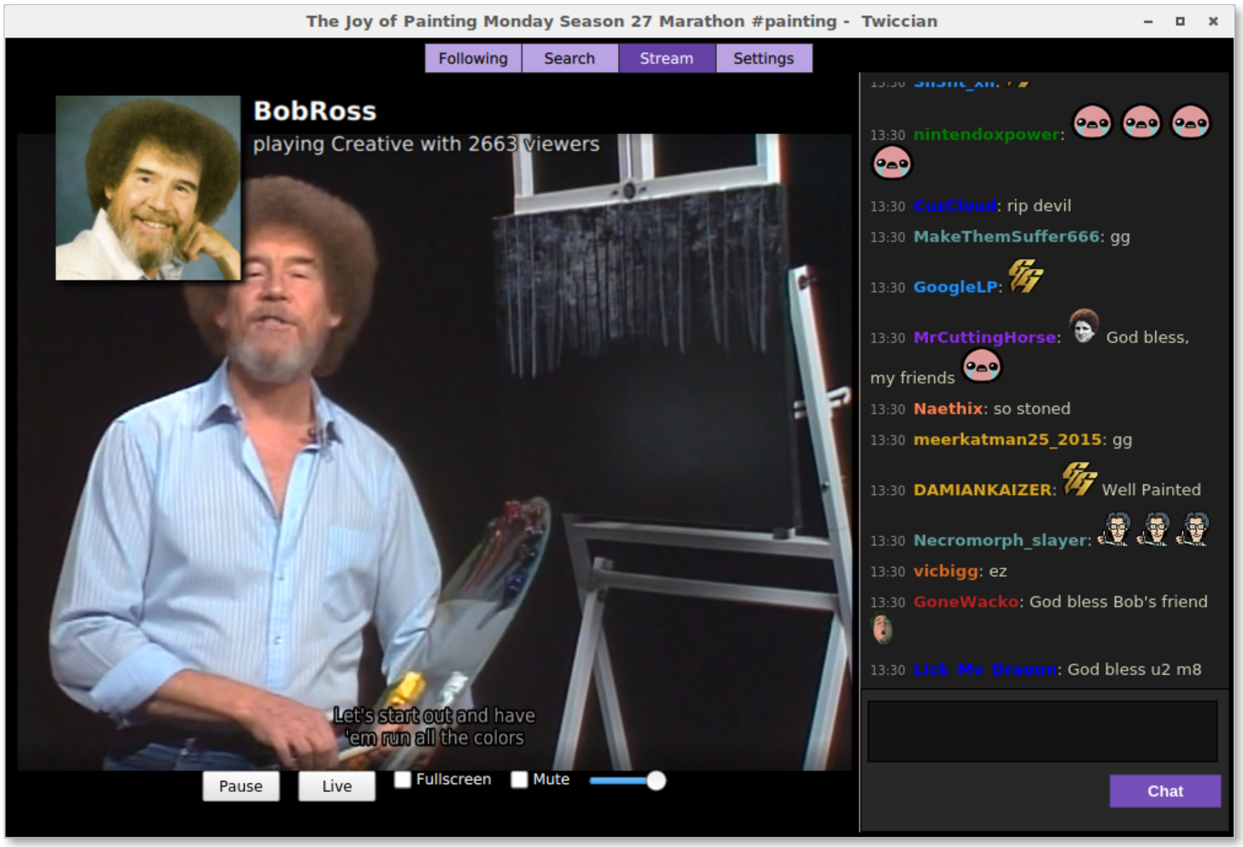
Super mario bros 3 rom hacking
Mitchflowerpower
playing Super Mario Bros. 3 with 56 viewers

UD3 - Races du lundi - Metal Slug X
UltimeDecathlon
playing Metal Slug X: Super Vehicle - 001 with 47 viewers

Super Mario 64 - 70 star speedrun

Figure 8: Results from a stream search

Now searching for a similar query from Search Games in Search Streams will return many more results [Figure 8]. This is due to what Twitch searches for in this general search. Where Search Games will search only for the currently played game, Search Streams searches in the: title, bio, recently played, and channel name. It can also attempt partial matches based on the search query. This search query was tweaked slightly more accurate results by using another suggested search call in the API.


The Joy of Painting Monday Season 27 Marathon #painting - Twiccian
Following
Search
Stream
Settings
BobRoss
playing Creative with 2663 viewers
Let's start out and have 'em run all the colors
Pause
Live
Fullscreen
Mute
Chat

13:30 nintendoxpower:
13:30 CuzCloud: rip devil
13:30 MakeThemSuffer666: gg
13:30 GoogleLP:
13:30 MrCuttingHorse: God bless, my friends
13:30 Naethix: so stoned
13:30 meerkatman25_2015: gg
13:30 DAMIANKAIZER: Well Painted
13:30 Necromorph_slayer:
13:30 vicbigg: ez
13:30 GoneWacko: God bless Bob's friend
13:30 Lick_My_Dragon: God bless u2 m8

Figure 9: Stream view with control overlay

Finally there is the Stream view, which largely stayed the same. Noticeable changes deal with the new information shown next to the avatar icon, which updates on every newly selected stream [Figure 9]. The title, already being in the title bar, was removed from the stream view to only show in the title bar. Much of this design is mimicked from Twitch's stream view on Chromecast. The other updates affect the chat view, with the aforementioned IRCv3 tagging and JavaScript updates. The emoticon cache is now reset on every launch to update any new emoticons, and also stores the badges once they are initially called. JavaScript also now stops auto-scrolling of new messages if the user scrolls far enough up the message history.

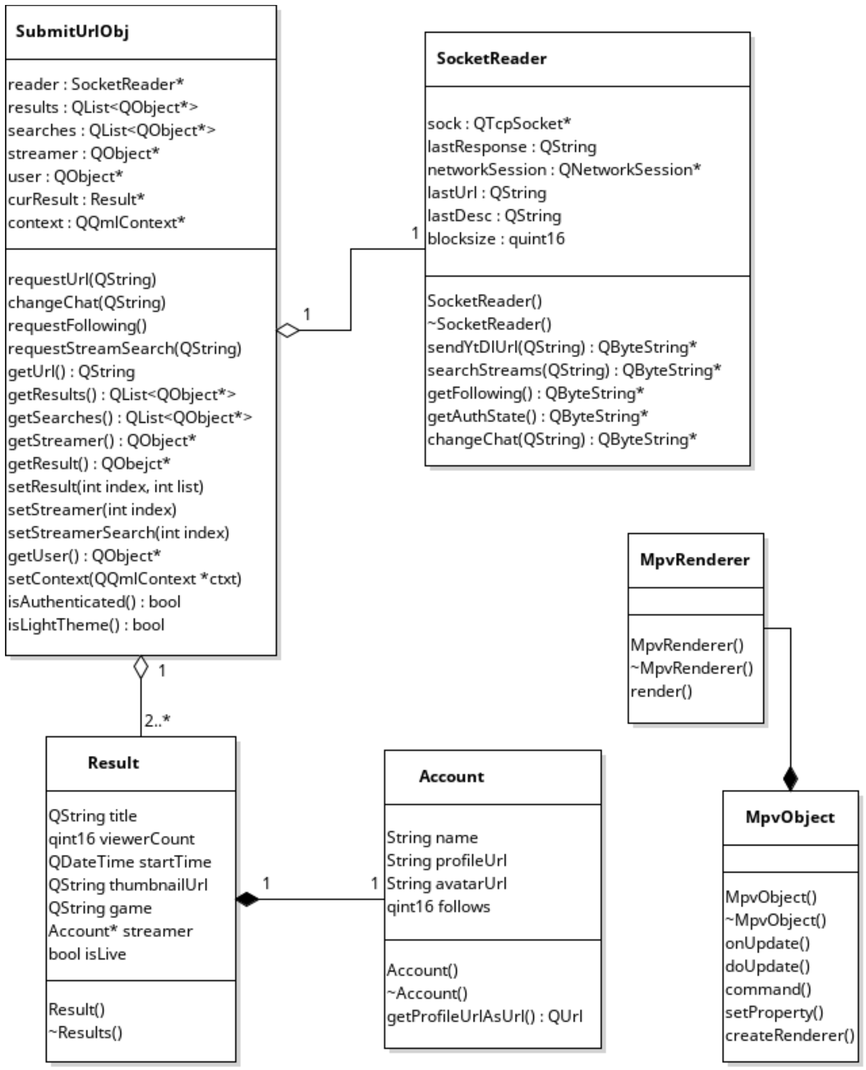


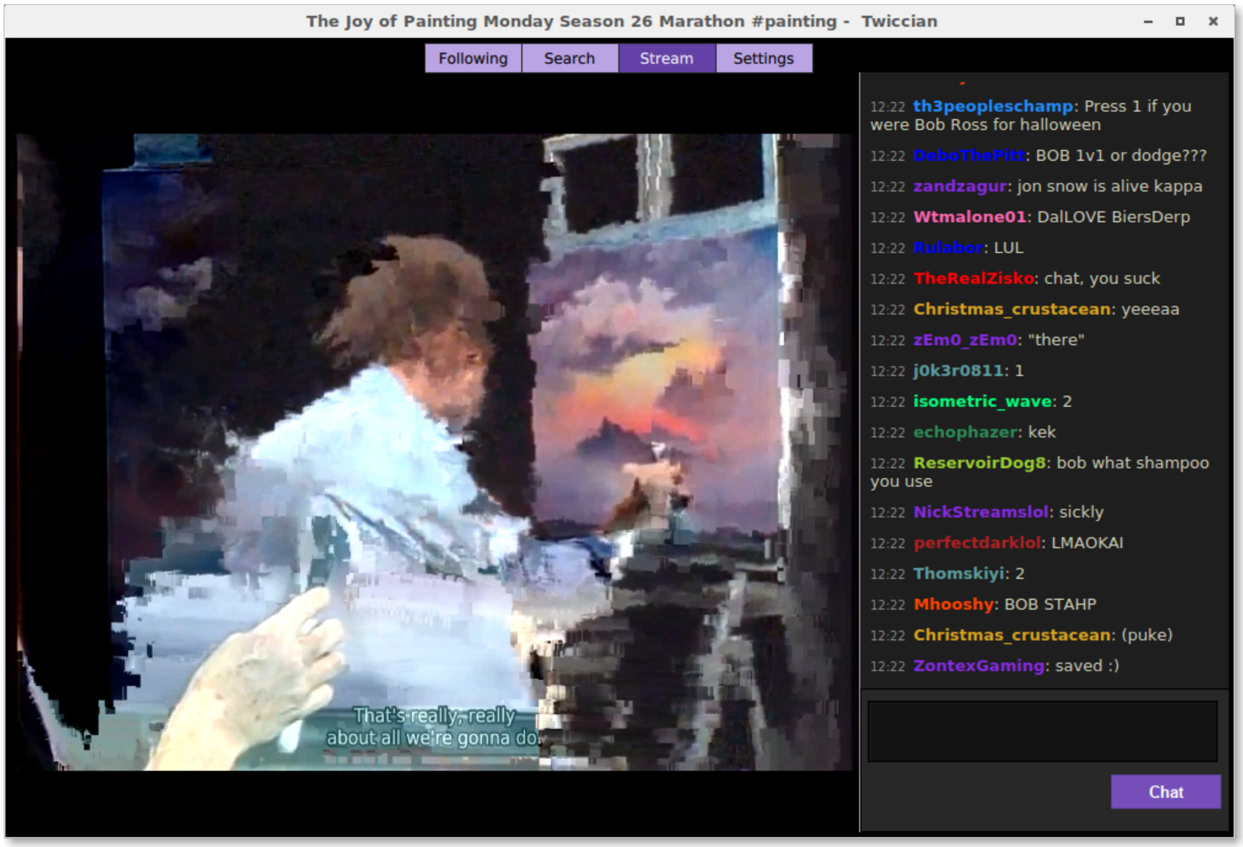
Figure 10: Complete UML diagram of the Twiccian client

4. Future Work

Any future work on Twiccian would result in simplifying the application from two applications to one [Figure 10]. The daemon would be dropped via the SocketReader, and all functionality moved to the frontend. This would allow for more features with less hassle, such as subscriber badges and more reliant searches. This would also be the optimal time to switch languages and packages for a more universal application.[8] Currently, mpv is limited to only Arch Linux platforms due to the development package

[8] "Using Qt with Alternative Programming Languages - Part 1"

available. The only part of the application that would stay would be Qt for the GUI, as QML itself is not a bad graphic language. This is all reliant on asynchronous calls to allow Twiccian to do more as a standalone application. It has also been noted that video quality selection could also be added. The current quality is set to the source quality, being whatever is selected by the live channel. Selecting a lower quality could allow for better performance on slow networks or weaker computers [Figure 11].


The Joy of Painting Monday Season 26 Marathon #painting - Twiccian
Following
Search
Stream
Settings
That's really, really about all we're gonna do.
Chat

12:22 th3peopleschamp: Press 1 if you were Bob Ross for halloween
12:22 DeboThePitt: BOB 1v1 or dodge???
12:22 zandzagur: jon snow is alive kappa
12:22 Wtmalone01: DalLOVE BiersDerp
12:22 Rulabor: LUL
12:22 TheRealZisko: chat, you suck
12:22 Christmas_crustacean: yeeeaa
12:22 zEm0_zEm0: "there"
12:22 j0k3r0811: 1
12:22 isometric_wave: 2
12:22 echophazer: kek
12:22 ReservoirDog8: bob what shampoo you use
12:22 NickStreamslol: sickly
12:22 perfectdarklol: LMAOKAI
12:22 Thomskiyi: 2
12:22 Mhooshy: BOB STAHP
12:22 Christmas_crustacean: (puke)
12:22 ZontexGaming: saved :)

Figure 11: Twiccian running on a slow network, chat indifferent

References

Demuzere, Vic, Go irc package, (2015), GitHub repository, https://github.com/sorcix/irc.

"IRCv3 Specifications." *IRCv3 Working Group*. 2016. Web.

"Justintv/Twitch-API." *GitHub*. 31 Aug. 2015. Web.

"PSA: Chat servers are going to migrate to AWS EC2 servers" *Twitch Developer Forums* 20 February 2016. Web.

"Qt QML 5.6." *Qt Documentation*. 2015. Web.

"Using Qt with Alternative Programming Languages - Part 1." *ICS*. 19 Aug. 2015. Web.