Implement cloud storages support in ScummVM

Contact information

Name: Alexander Tkachev aka Tkachov
Email: alexander@tkachov.ru
Phone: 89232242789
Timezone: UTC +6

Idea

The idea is to implement cloud support in ScummVM. Users would be able to export their save files and game data in different cloud storage services and then import those on other devices.

At least these cloud services would be supported:

- Dropbox;
- Box;
- OneDrive;
- Google Drive.

All of them use OAuth2 and provide an API to manage user's files.

Things to be implemented:

- portable (using libcurl) API interaction for all supported cloud services;
- saves sync feature;
- downloading games from cloud feature;
- uploading ScummVM-detected games into the cloud feature;
- GUI for user to interact with.

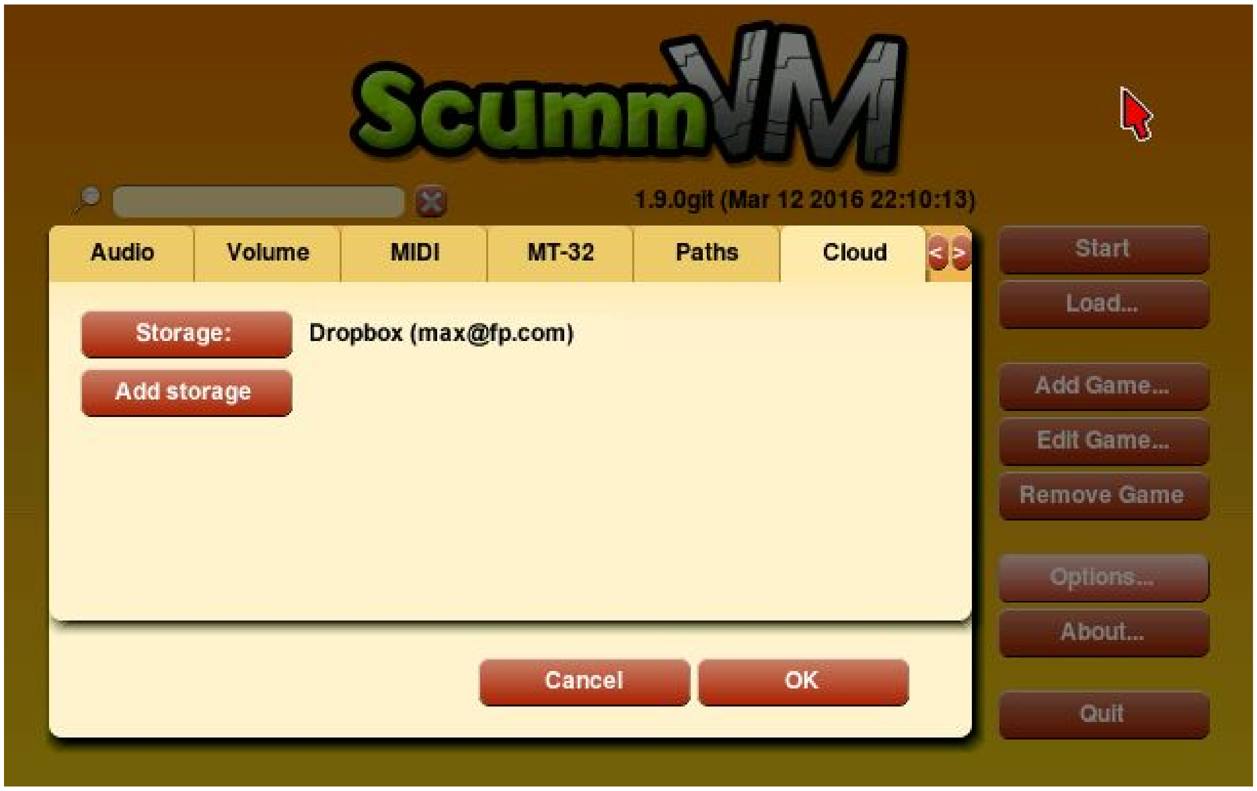
Extra tasks:

- download games in background, so users can play other games while downloading;
- URL opening (for OAuth), so users can click the link instead of typing it in the browser;
- local webserver (for OAuth), so users won't have to type OAuth code from the browser;
- allow users to download freeware game data from ScummVM (scummvm.org/games).

Possible mentors: Eugene Sandulenko, Peter Bozso.

Detailed feature description

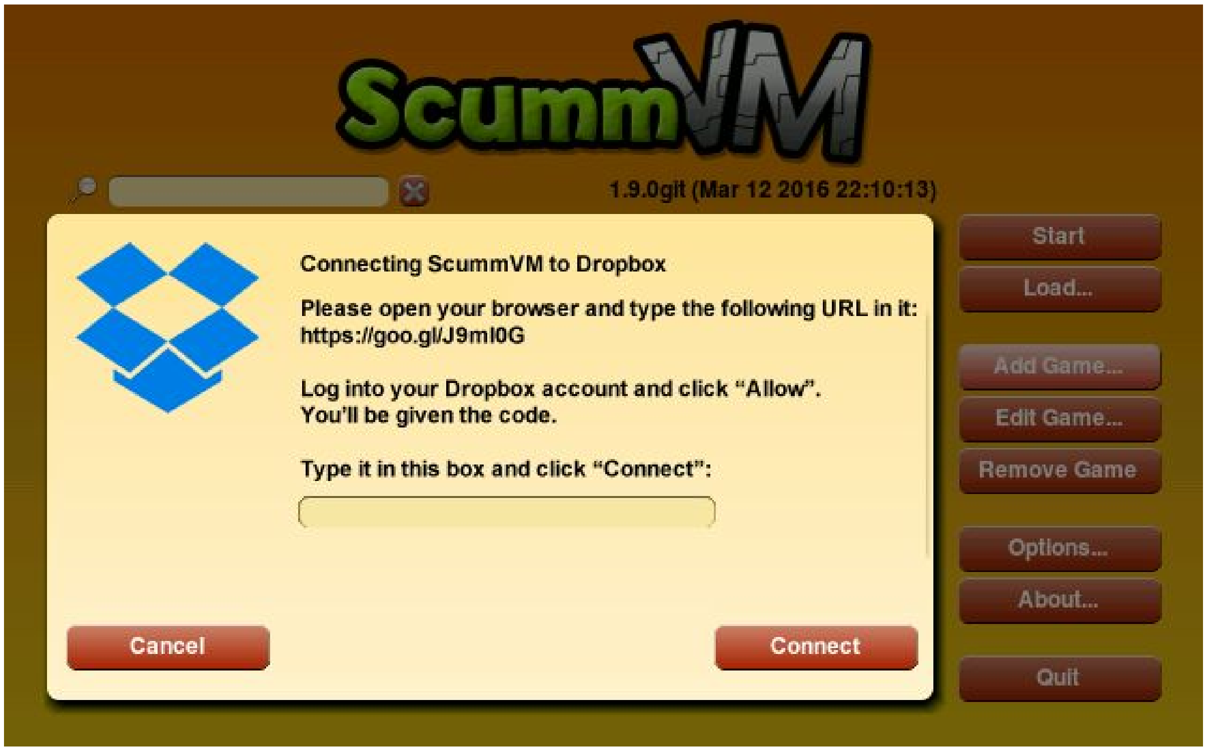
User would be able to connect cloud storages on Cloud tab of Options menu:



Storage: Dropbox (max@fp.com)

Only one storage can be selected as active. Other connected storages are remembered, so user can switch between them easily.

When user selects which cloud storage provider he wants to add, he/she is asked to open a browser and type a URL into it. This is shortened URL which redirects user to cloud provider's OAuth page.



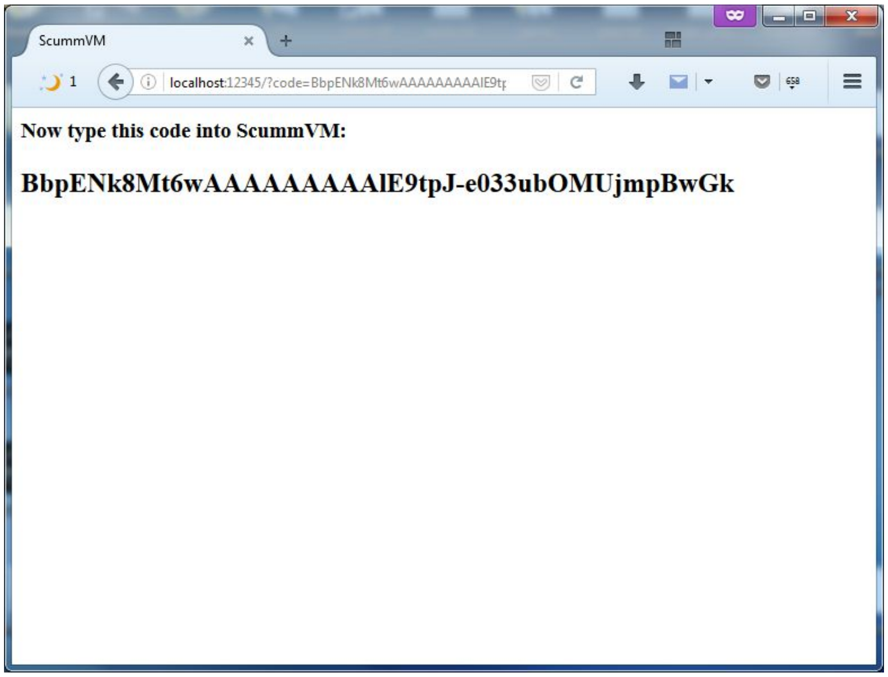
Connecting ScummVM to Dropbox

Please open your browser and type the following URL in it:
https://goo.gl/J9ml0G

Log into your Dropbox account and click "Allow".
You'll be given the code.

Type it in this box and click "Connect":

User must press the "Allow" button in order to get the code for ScummVM. When he/she does, browser is redirected to the ScummVM https:// page with the code passed in the URL:



localhost:12345/?code=BbpENk8Mt6wAAAAAAAAAlE9tp

Now type this code into ScummVM:

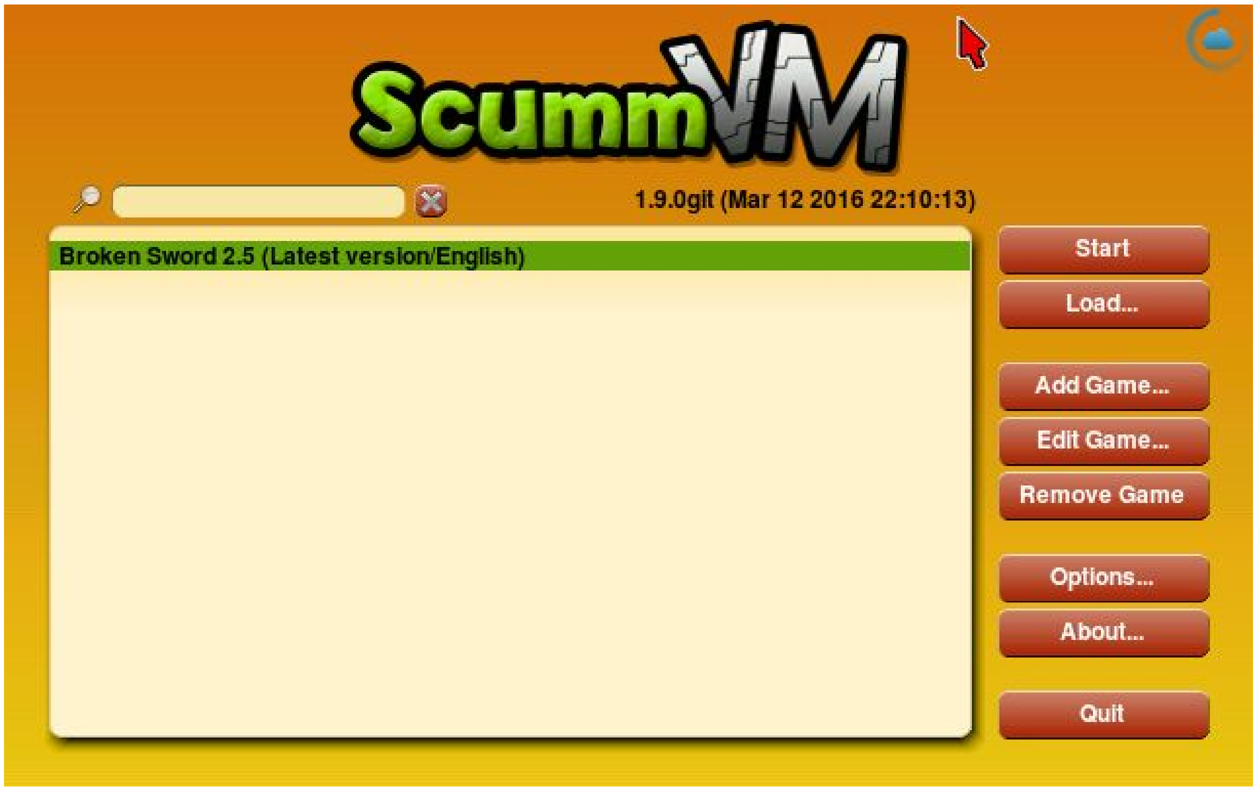
BbpENk8Mt6wAAAAAAAAAlE9tpJ-e033ubOMUjmpBwGk

Simple PHP or JS can extract the code from URL and put it on the page (as shown above). This redirection page is either https:// or localhost. I suggest we use https:// link. I might be able to install SSL-certificate for scummvm.org, though I think that must be done by site's webmaster.

localhost might be used when ScummVM runs local webserver, but as that might be too complicated, I suggest me implementing it only as extra task (if there is time left, that is).

So, user types this code into that input box and presses "Connect" button. ScummVM makes an HTTP request using libcurl and gets the access token by showing the code. After that, ScummVM uses token to make API requests.

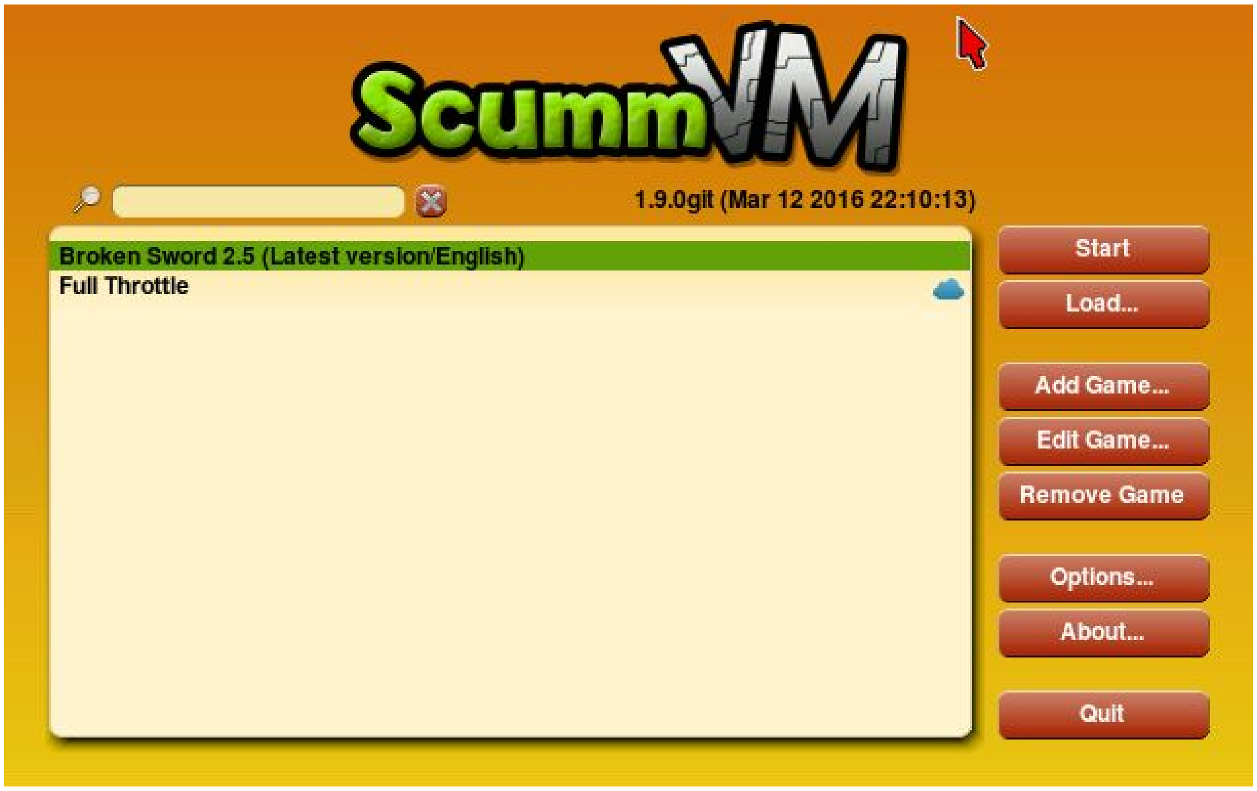
Once connected, ScummVM would sync user's local save games with remote cloud storage "Saves" folder. The newest files replace the older, if there are any conflicts. While syncing, small sync icon would be shown in the corner:



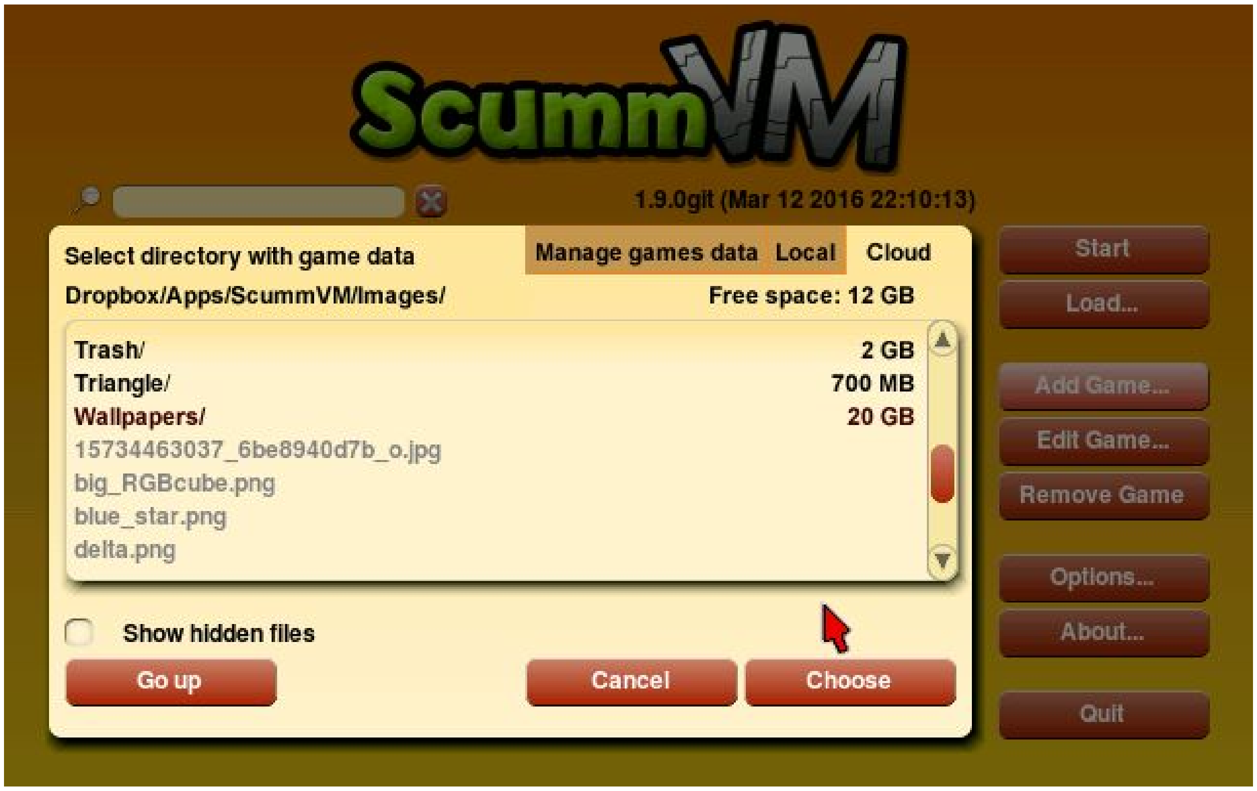
Saves are synced when user changes active cloud storage, launches save/load dialog or when autosave is created.

Saves are synced automatically into the "Saves" folder on cloud storage. It's kept in "ScummVM" app folder (visible for user) and this path cannot be changed. Still, user can access his saves through cloud storage interface.

Only downloaded games are shown in the list. Games downloaded from cloud are marked with cloud icon:



"Add Game..." dialog is extended with "Cloud" tab, where user can select a directory to download. In the same dialog the user sees size of every game as well as amount of free space on their device. Games which will not fit are colored dark red:



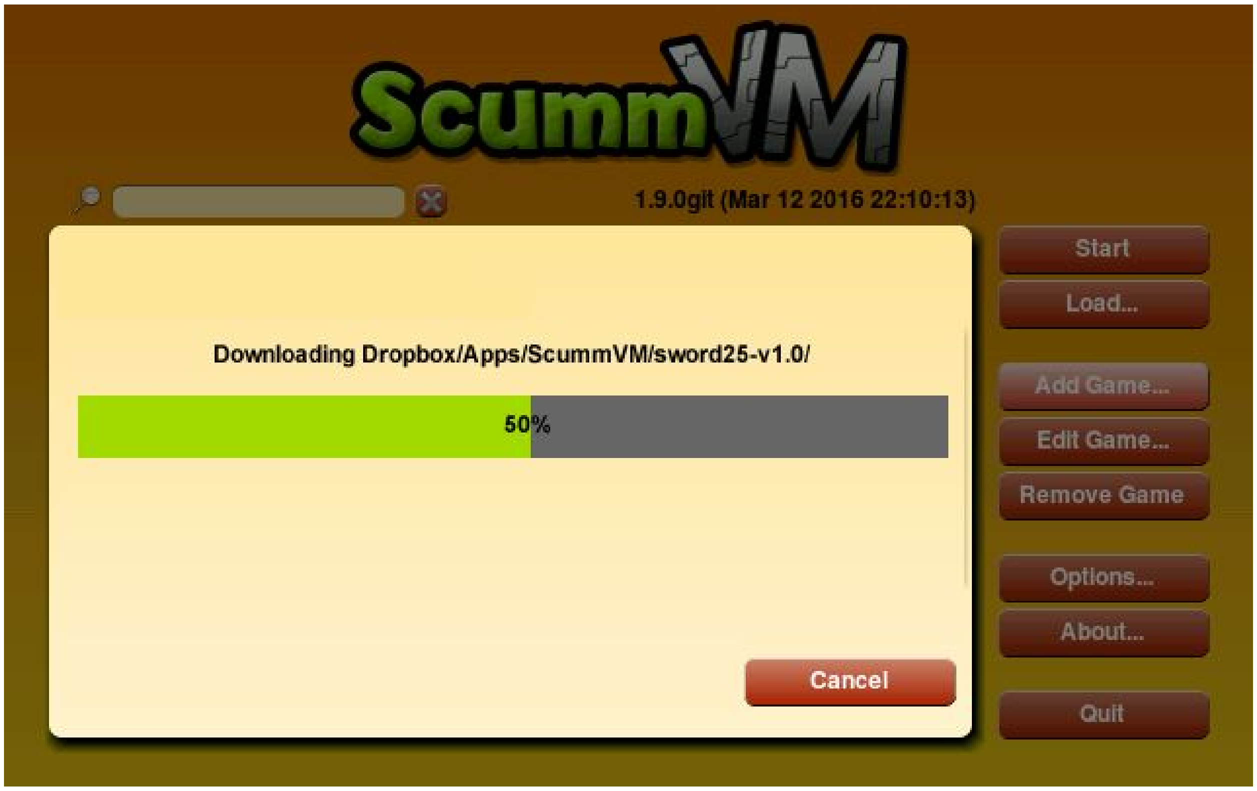
Select directory with game data

Dropbox/Apps/ScummVM/Images/ Free space: 12 GB

Trash/	2 GB
Triangle/	700 MB
Wallpapers/	20 GB
15734463037_6be8940d7b_o.jpg	
big_RGBcube.png	
blue_star.png	
delta.png	

If user is on cellular network, the message would appear asking user to confirm that he wants to download the game.

Users can't play ScummVM games during the download. They can abort the process using "Cancel" button:

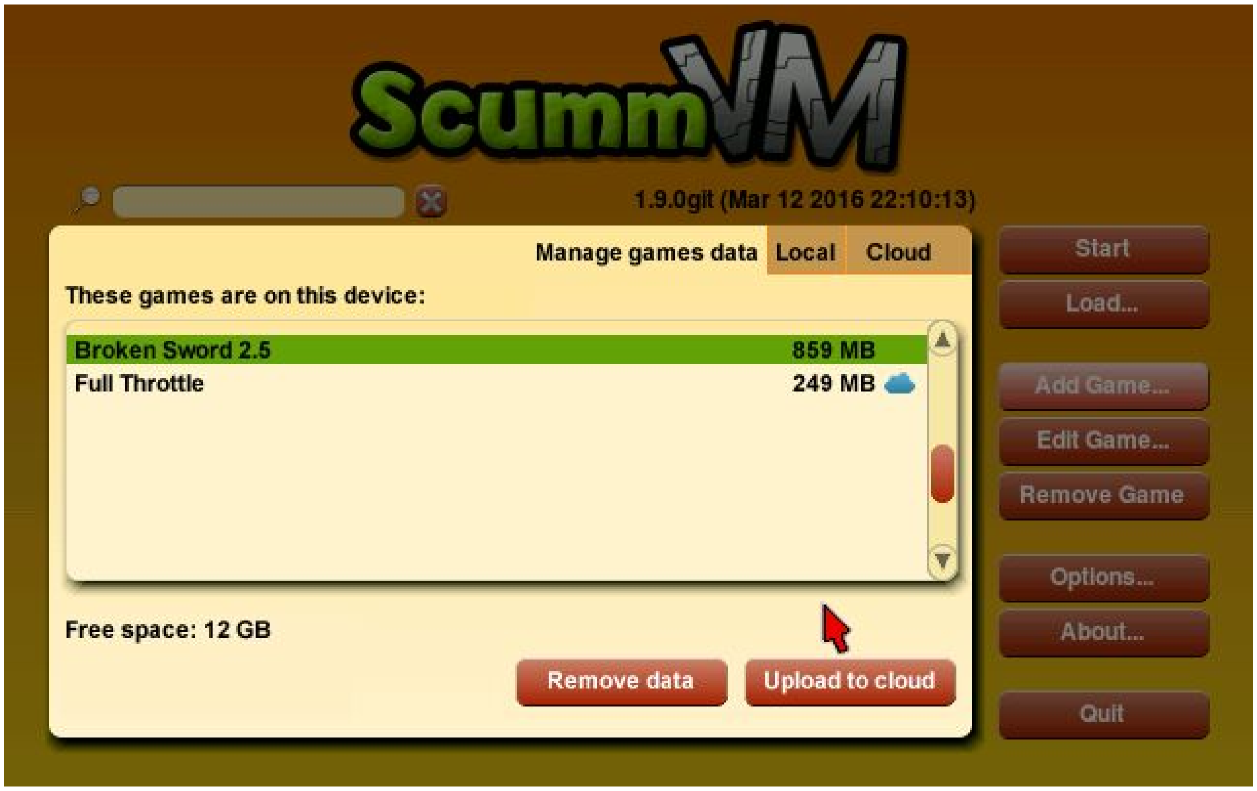


Downloading Dropbox/Apps/ScummVM/sword25-v1.0/

When the game is downloaded, ScummVM runs game detection procedure. If no game is detected, user is asked whether he wants to remove the data.

Extra challenge: do downloading in the background and forbid using this directory until it's done. Give users the ability to play other games during the download.

"Manage games data" tab is added in Options menu (or as a separate button in the launcher menu). It opens a list of all games on local device. User can remove game data with "Remove data" button and upload the game into the cloud with "Upload to cloud" button:



These games are on this device:

Broken Sword 2.5	859 MB
Full Throttle	249 MB

Free space: 12 GB

If the game was not downloaded from the cloud, users are asked whether they really want to kill the data. Uploaded games folders would also contain a metafile for ScummVM to know which game was detected within that folder.

OAuth and API usage experience

I've been working with OAuth and different APIs in the past. I've made a simple application which uses Dropbox API via libcurl. It also includes some pthreads and socket usage and demonstrates the idea of using local webserver to get the code from the service without user copy-pasting it into the application.

Timeline

Before coding	
Study services API and ScummVM code (GUI and settings storaging). I probably will make a test prototype application which is using API of these services in order to get some experience. That will help me design common interface for these APIs well.	
May 23 — June 28	
May 23 — May 29	Write Storage interface, implement tokens saving. Add Dropbox and OneDrive storages.
May 30 — June 5	Implement file downloading (Dropbox and OneDrive). Add auto detection procedure running after download.
Milestone #1: first ScummVM-downloaded cloud game	
June 6 — June 12	Implement saves sync (Dropbox and OneDrive): ● do sync on first connect, save/load dialog or autosave; ● keep them in "Saves" folder and replace with newest on all conflicts.
June 13 — June 19	Implement Box support
Midterm (June 21 — June 28)	
June 20 — June 26	Implement Google Drive support
Milestone #2: sync and downloading works at least with Dropbox and OneDrive	
June 28 — August 16	
June 27 — July 3	Add Cloud tab in Options menu. Add "Add storage" dialog. Show selected storage and make storage selection feature.
Milestone #3: 4 storages supported	
July 4 — July 10	Cellular network API. Add Cloud tab in "Add Game..." dialog and directory selector.

	Add downloading dialog with "Cancel" button and progress bar. Ask before downloading big games using cellular network. Ask if user wants to remove data if detection failed.
July 11 — July 17	Finish up cloud directory selector: ● add sizes of directories (red when won't fit); ● show amount of free space on device. Add some flag for games to show if was downloaded from cloud and show such games in the list with cloud icon. Show icon during saves syncing.
July 18 — July 24	Add "Manage game data" tab: ● list all local games with it's size and cloud icon if it was downloaded from cloud; ● "Remove data" button (asks twice whether user wants to remove non-cloud game); ● "Upload to cloud" button (adds metafile into cloud storage; has it's own uploading dialog).
Milestone #4: GUI added	
July 25 — July 31, August 1 — August 7, August 8 — August 14	Time to do extra tasks, polish everything and fix bugs.

I must add that I'm having my exams in June, so I might work a little less at that time. I've had my exams in June last year during GSoC 2015 and had no problems with finishing the project. I'm able and willing to work more than 40 hours a week (with no weekends, for example) to catch up with my schedule.

Biographical information

I'm a third year undergraduate at the Department of Information Technology of Novosibirsk State University.

I've already participated in GSoC in 2015 and successfully finished my project (C++/Céu, SDL2).

I've also been participating in Parallels Summer School 2014 (we were developing a multiplayer game for Oculus Rift).

I don't have any open source experience other than GSoC 2015. I have a project on Github (C++/SDL2/OpenGL layer that can be used as simple game engine, works on Windows, Ubuntu and Android), but I'm not sure it counts as open source experience.

I've been developing simple games for years now. I used different engines like GameMaker and Unity and languages like C#, C/C++, ActionScript 3. I usually develop games on my own, but I also worked in a team sometimes. Have some experience with git. Some of my projects are available on my site.

I've fixed ScummVM bug #6980 and sent it as pull request #689. It is already merged into master.

I take part in different contests like ACMs and CTFs, personally or with my friends as a team. Sometimes we have quite good results. A few years ago we've been writing AI for bots on Lua and we took 2nd place. We took 1st place on two Attack/Defence competitions in Russia in 2014: SibirCTF and KrasCTF. We won SibirCTF and KrasCTF in 2015 again. We have also been to Paris to compete in Nuit du Hack 2k15 and we're going to visit finals of RuCTF 2016.

I've been working in team writing an ActionScript 3 game for social networks using git as VCS. During Parallels Summer School 2014 we were developing an Oculus Rift game using Unity as a team, but no VCS was used. During Parallels Winter School 2015 we've been developing a game using Qt and OpenGL, private Github repository was used.

I also have developed a few Android applications (using both Java and C/C++ NDK) and even have one published in Google Play. I had some experience with Qt and C# (Qt Develop and Visual Studio, Unity). My sites are written on HTML/CSS/JS/PHP. I write Python scripts sometimes.