Deon Zhao
Harry Duong
Thuc Nguyen

Runelite - Architecture and Social Context

In order to document the architecture for Runelite, we first went on Runelite's official Discord channel and asked in the #support channel whether an architecture has already been documented for this system. We received a response from the user "dkvl" that another user "abex" previously designed an architecture for Runelite (*Figure 1*).

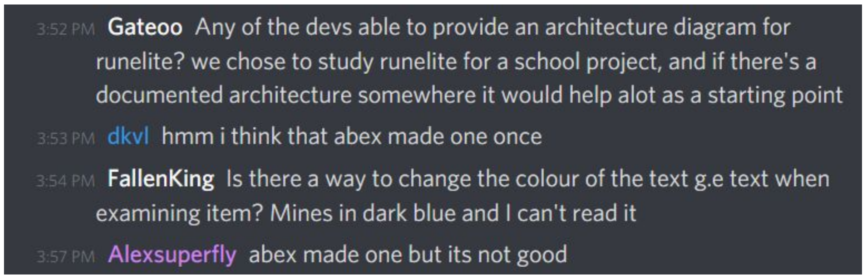
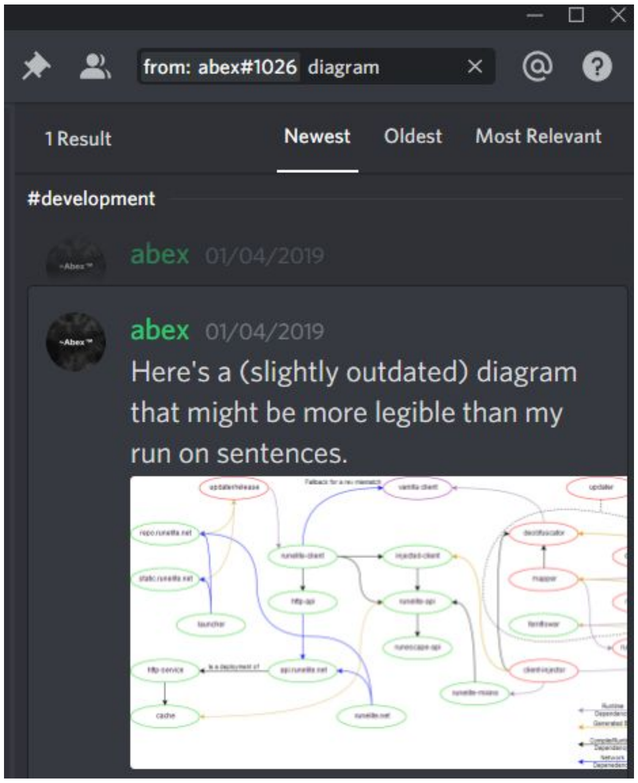

3:52 PM Gateoo Any of the devs able to provide an architecture diagram for runelite? we chose to study runelite for a school project, and if there's a documented architecture somewhere it would help alot as a starting point
3:53 PM dkvl hmm i think that abex made one once
3:54 PM FallenKing Is there a way to change the colour of the text g.e text when examining item? Mines in dark blue and I can't read it
3:57 PM Alexsuperfly abex made one but its not good

***Figure 1**: Deon (Gateoo) asked for an existing architecture on Runelite's discord channel, directing us to user "abex"*

We searched the chat history of user "abex" for the keyword diagram in hopes of finding any mention of the architecture. This search successfully led us to a diagram of Runelite's architecture as written by abex.

The diagram we found is shown on the right

A better view of the diagram is shown below, in *Figure 2*.



abex 01/04/2019
Here's a (slightly outdated) diagram that might be more legible than my run on sentences.

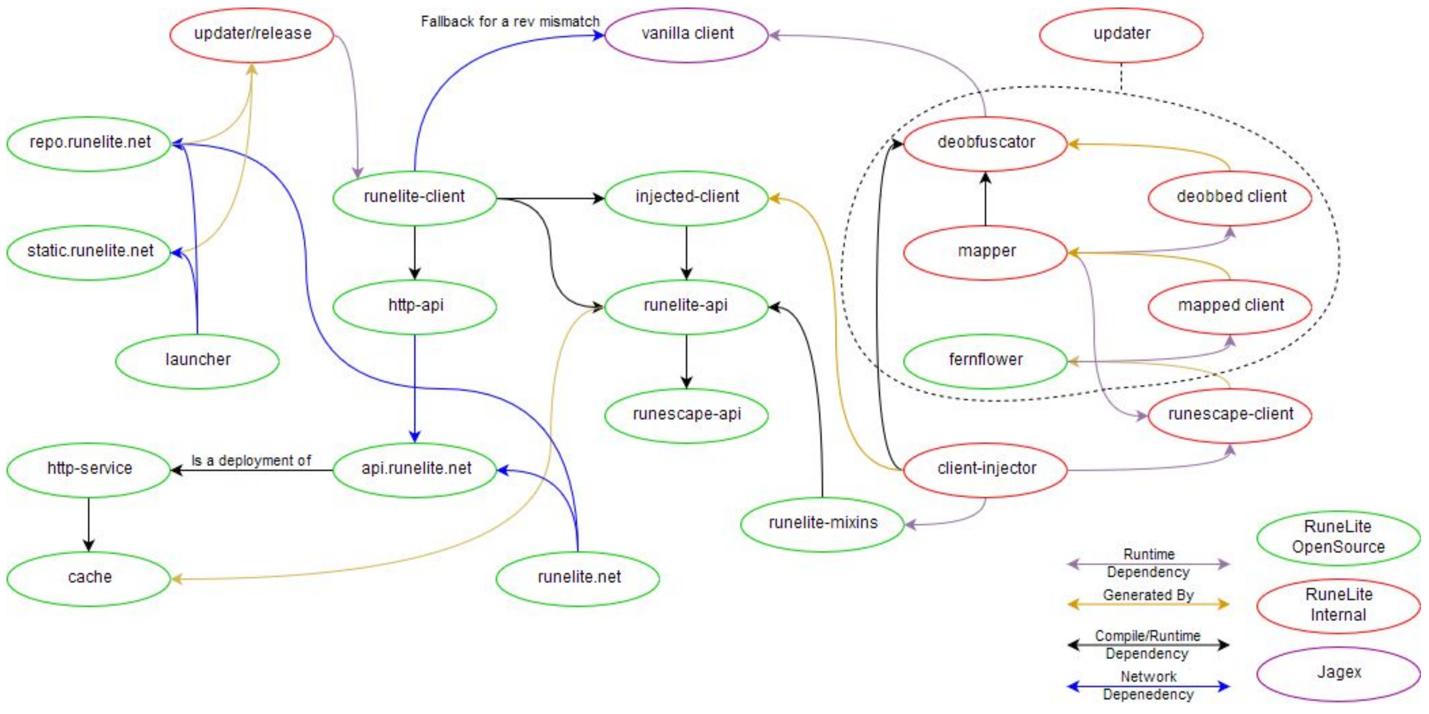
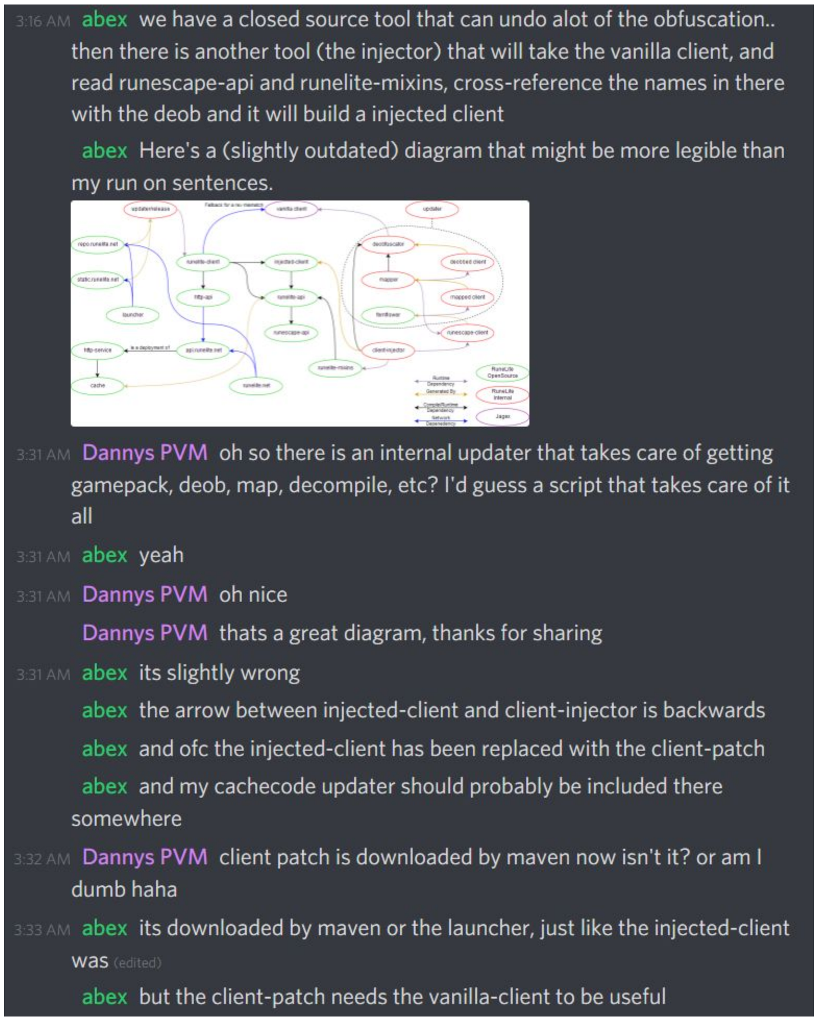


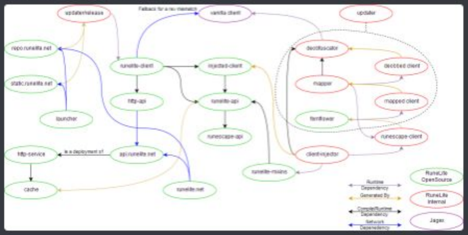
Figure 2*: A diagram depicting the architecture for Runelite. Runelite opensource, bordered green, are components of the open source system made available on Github. Runelite Internal, bordered red, deals with extracting game data from the Jagex client. Jagex, bordered purple, is the vanilla client provided by Runescape.*

The diagram in *Figure 2* separated the architecture of Runelite mainly by their packages. Although outdated, we see that some of these components such as launcher, http-service, cache, runelite-client, and http-api matches either the package names or class names on Github. Because it is outdated, some packages or missing, renamed, or updated which meant that this diagram is no longer in sync with the as-implemented architecture of Runelite. However, It was still helpful because the diagram showed us how Runelite interacts with both the vanilla client and the internal packages and classes. These internal components which are closed source, communicate with the vanilla client to complete tasks that include extracting game data, using the Runescape API, and mapping names with a deobfuscator component. The chat that abex had with another user (*Figure 3*) is the information we found that clarified the outdated architecture.



3:16 AM abex we have a closed source tool that can undo alot of the obfuscation.. then there is another tool (the injector) that will take the vanilla client, and read runescape-api and runelite-mixins, cross-reference the names in there with the deob and it will build a injected client

abex Here's a (slightly outdated) diagram that might be more legible than my run on sentences.



3:31 AM Dannys PVM oh so there is an internal updater that takes care of getting gamepack, deob, map, decompile, etc? I'd guess a script that takes care of it all

3:31 AM abex yeah

3:31 AM Dannys PVM oh nice

Dannys PVM thats a great diagram, thanks for sharing

3:31 AM abex its slightly wrong

abex the arrow between injected-client and client-injector is backwards

abex and ofc the injected-client has been replaced with the client-patch

abex and my cachecode updater should probably be included there somewhere

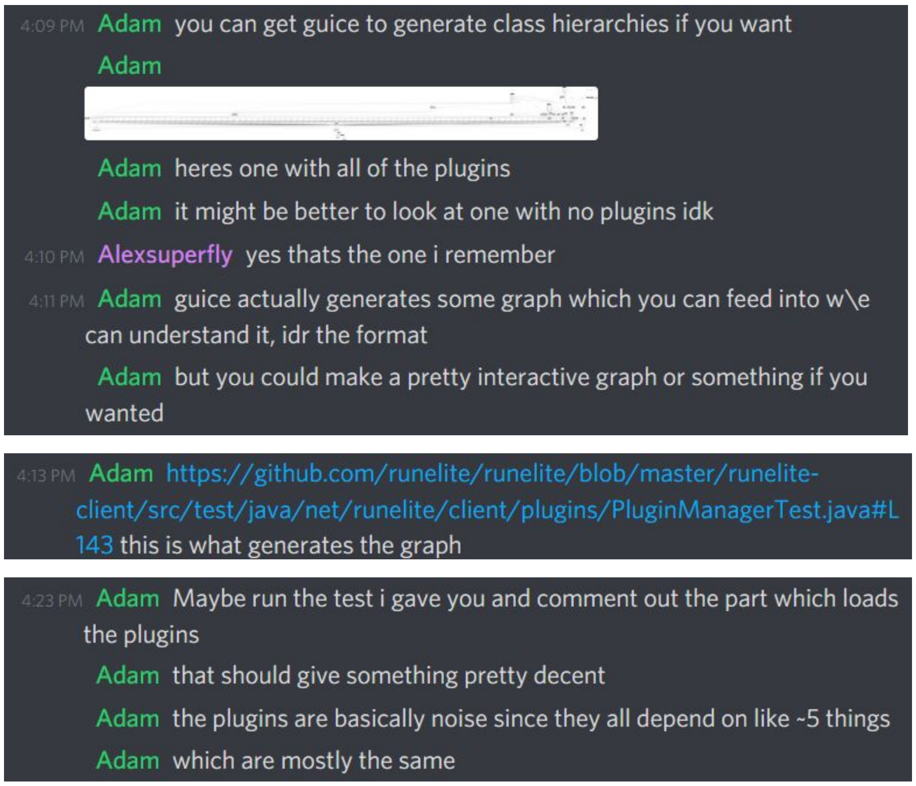
3:32 AM Dannys PVM client patch is downloaded by maven now isn't it? or am I dumb haha

3:33 AM abex its downloaded by maven or the launcher, just like the injected-client was (edited)

abex but the client-patch needs the vanilla-client to be useful

Figure 3*: The chat history between "abex" and "Dannys PVM" detailing parts of the diagram. At that time, abex notes that the diagram was already out of date.*

Our next step was to find a way to generate a diagram that can illustrate the as-implemented architecture of Runelite. Deon asked Adam (the creator of Runelite) on Discord about how we can obtain some sort of as-implemented architecture and he suggested that we generate class hierarchies without the plugins. This made sense because plugins are independent and can individually and dynamically be added to the system. Taking out these extra layers would help us see what a possible architecture for Runelite would currently look like. Adam also suggested that we look at one of the test files, called PluginManagerTest and comment out the code block that loads plugins so that we can generate a graph that can provide a perspective on the architecture. We did so and found a dumpGraph() method which could generate a graph and output it to a specified path. Deon's chat with Adam is shown in *Figure 4* below, and the dumpGraph() method in PluginManagerTest.java is shown in *Figure 5*.



4:09 PM **Adam** you can get guice to generate class hierarchies if you want

Adam

Adam heres one with all of the plugins

Adam it might be better to look at one with no plugins idk

4:10 PM **Alexsuperfly** yes thats the one i remember

4:11 PM **Adam** guice actually generates some graph which you can feed into w\e can understand it, idr the format

Adam but you could make a pretty interactive graph or something if you wanted

4:13 PM **Adam** https://github.com/runelite/runelite/blob/master/runelite-client/src/test/java/net/runelite/client/plugins/PluginManagerTest.java#L143 this is what generates the graph

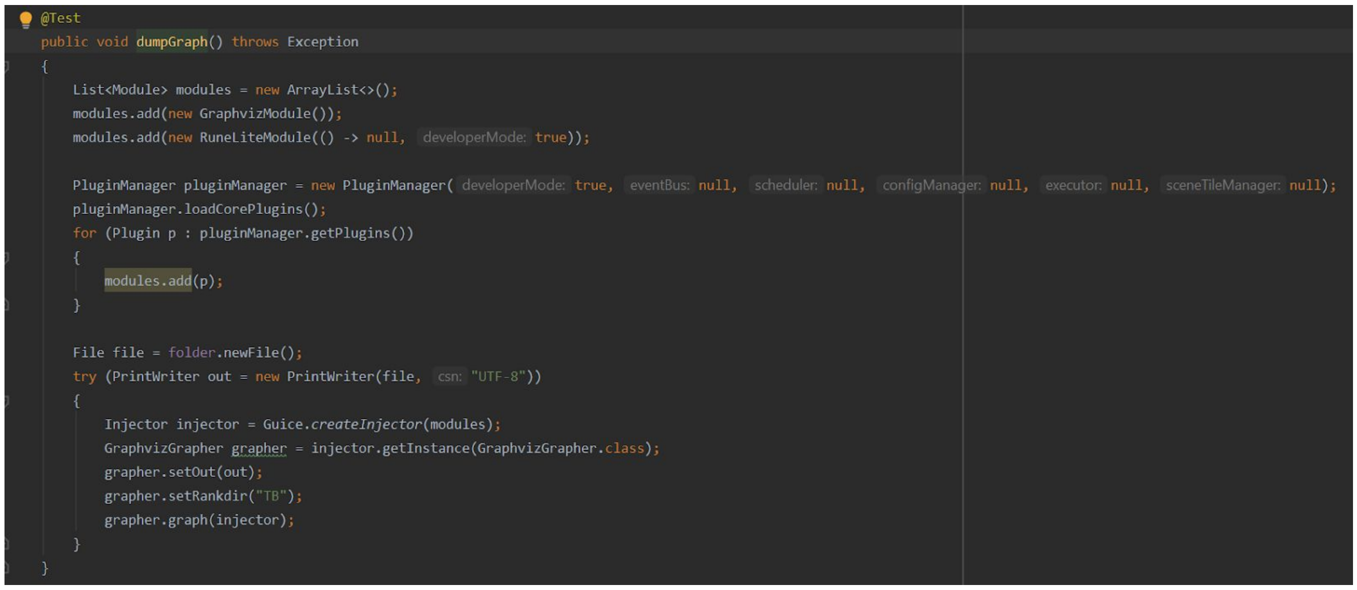
4:23 PM **Adam** Maybe run the test i gave you and comment out the part which loads the plugins

Adam that should give something pretty decent

Adam the plugins are basically noise since they all depend on like ~5 things

Adam which are mostly the same

Figure 4*: Deon's chat with Adam about how we can generate some sort of graph or diagram for the architecture of Runelite. Adam directs us to the test file called PluginManagerTest.*



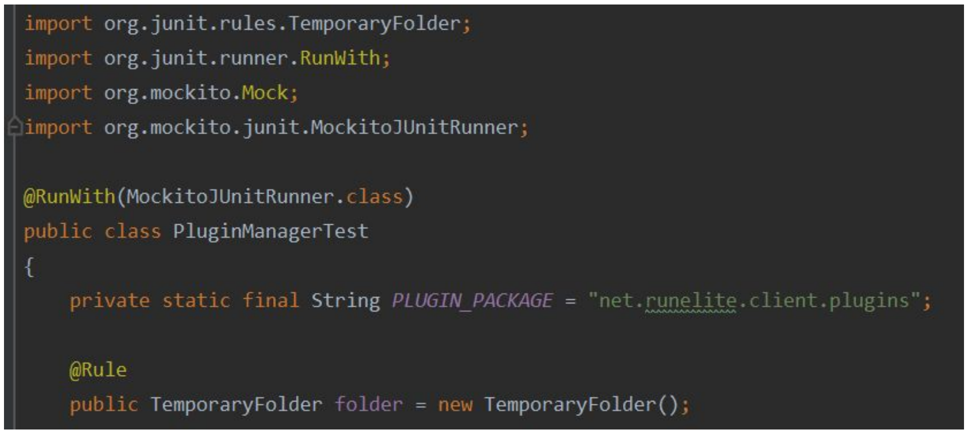
```
@Test
public void dumpGraph() throws Exception
{
    List<Module> modules = new ArrayList<>();
    modules.add(new GraphvizModule());
    modules.add(new RuneLiteModule(() -> null,  developerMode: true));

    PluginManager pluginManager = new PluginManager( developerMode: true,  eventBus: null,  scheduler: null,  configManager: null,  executor: null,  sceneTileManager: null);
    pluginManager.loadCorePlugins();
    for (Plugin p : pluginManager.getPlugins())
    {
        modules.add(p);
    }

    File file = folder.newFile();
    try (PrintWriter out = new PrintWriter(file,  csn: "UTF-8"))
    {
        Injector injector = Guice.createInjector(modules);
        GraphvizGrapher grapher = injector.getInstance(GraphvizGrapher.class);
        grapher.setOut(out);
        grapher.setRankdir("TB");
        grapher.graph(injector);
    }
}
```

Figure 5*: The dumpGraph() method in PluginManagerTest.java which generates a graph by using the GraphvizModule, RuneliteModule, Injector, and GraphvizGrapher classes, and outputting it into a file.*

Taking a more thorough look at PluginManagerTest, we can see that the dumped graph is being saved into a variable called folder (*Figure 6*). The folder variable is then found to be of type TemporaryFolder generated during runtime by JUnit and deleted when the test finishes running. To save the actual dumped graph we had to modify the code to save it where we want it to be (*Figure 7*). In other words, within the dumpGraph() method, we commented out the statement that referenced the TemporaryFolder and replaced it with a new File object so that we can actually access the file and its contents.

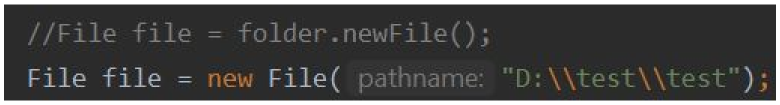


```
import org.junit.rules.TemporaryFolder;
import org.junit.runner.RunWith;
import org.mockito.Mock;
import org.mockito.junit.MockitoJUnitRunner;

@RunWith(MockitoJUnitRunner.class)
public class PluginManagerTest
{
    private static final String PLUGIN_PACKAGE = "net.runelite.client.plugins";

    @Rule
    public TemporaryFolder folder = new TemporaryFolder();
```

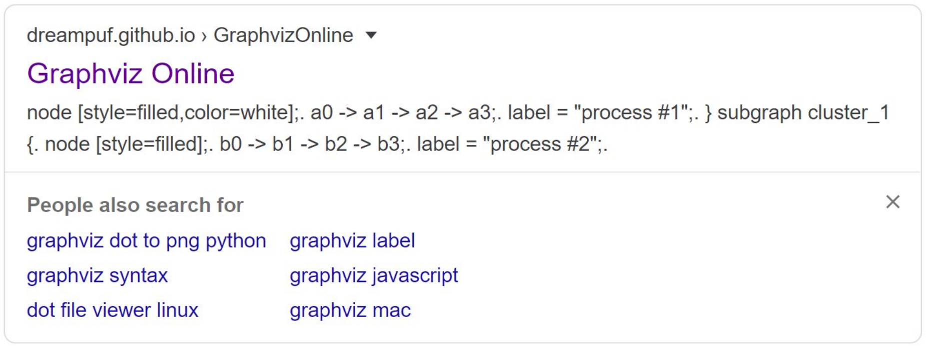
Figure 6*: In the PluginManagerTest class, the folder variable is where the generated graph will be saved to.*



```
//File file = folder.newFile();
File file = new File( pathname: "D:\\test\\test");
```

Figure 7*: The File object requires a path, which we must change.*

We decided to save it in the D drive in a folder called test, and named the dumped graph test as well. Not knowing what type of file was dumped, we decided to look for clues in the code. Earlier within the dumpGraph() method, we saw that an object of type GraphvizGrapher was responsible for writing the graph to a file. GraphvizGrapher is a class from the Guice API but what exactly is Graphviz? A quick google search of “Graphviz Viewer” returned this promising result (*Figure 8*). Further searching allowed us to understand that Graphviz is an online graph visualization tool that we can use to map structure to a diagram of graphs consisting of nodes and edges. It has its own text language to format these diagrams. We also looked at the website, and we can tell that the left side shows some input code, and the right side displayed a graph to output (*Figure 9*).



dreampuf.github.io › GraphvizOnline

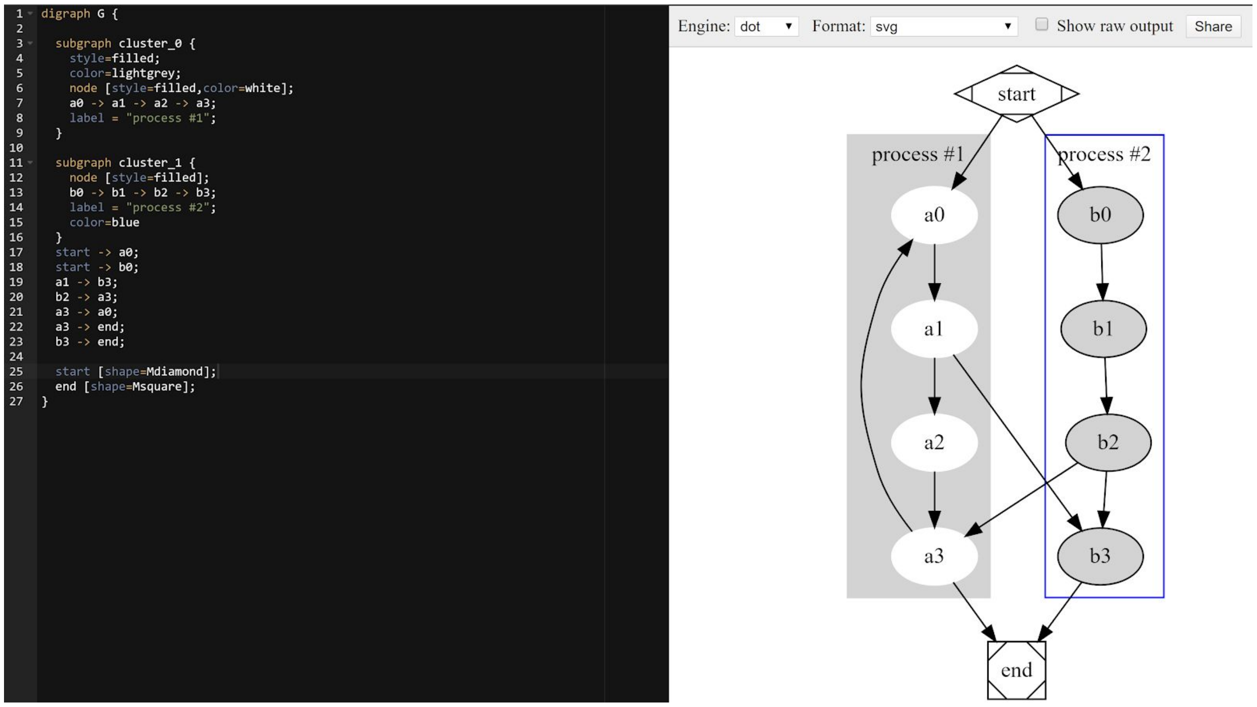
Graphviz Online

node [style=filled,color=white];. a0 -> a1 -> a2 -> a3;. label = "process #1";. } subgraph cluster_1 {. node [style=filled];. b0 -> b1 -> b2 -> b3;. label = "process #2";.

People also search for

graphviz dot to png python
graphviz label
graphviz syntax
graphviz javascript
dot file viewer linux
graphviz mac

Figure 8: Google search results for "GraphvizGrapher



```
digraph G {

  subgraph cluster_0 {
    style=filled;
    color=lightgrey;
    node [style=filled,color=white];
    a0 -> a1 -> a2 -> a3;
    label = "process #1";
  }

  subgraph cluster_1 {
    node [style=filled];
    b0 -> b1 -> b2 -> b3;
    label = "process #2";
    color=blue
  }
  start -> a0;
  start -> b0;
  a1 -> b3;
  b2 -> a3;
  a3 -> a0;
  a3 -> end;
  b3 -> end;

  start [shape=Mdiamond];
  end [shape=Msquare];
}
```

Engine: dot Format: svg Show raw output

Figure 9*: GraphvizGrapher online tool (*[*https://dreampuf.github.io/GraphvizOnline/*](https://dreampuf.github.io/GraphvizOnline/)*) which translates the text language on the left into a graph consisting of nodes and edges on the right.*

By opening up the dumped file with a text editor, we found out that it is in the exact format as the website's example, so we pasted it in on the website to get our graph. It came out exactly like Adam showed me in discord. Pretty much unreadable (*Figure 10*).

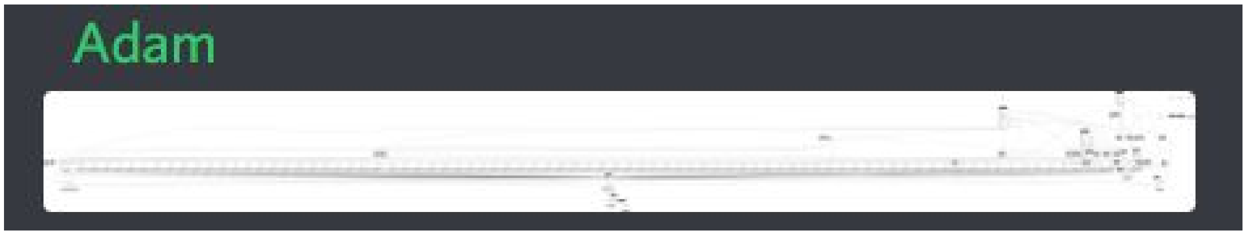
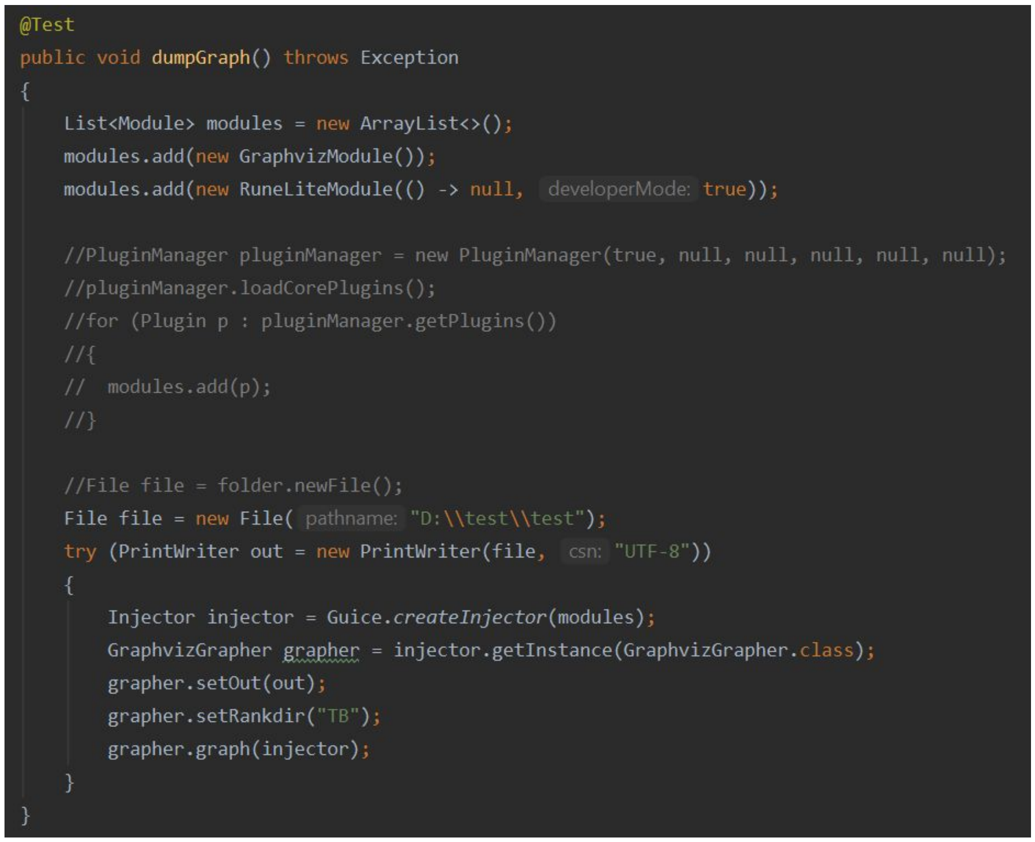


Figure 10*: The unreadable diagram of Runelite's architecture that Adam mentioned because of all the packages under plugins.*

We followed Adam's suggestion and commented out the part that loads in all the plugins, since they all depend on the same 5 things and just generate noise. The code below shows the statements that were commented out in the hopes of generating a simpler and a more focused diagram (*Figure 11*). We obtained successful results and were able to recover the as-implemented architecture as seen by the diagram in *Figure 12*. Looking at the diagram, we see that it displays the actual architecture of the Runelite client. The diagram made by abex showed the interactions between Runelite with Runelite internal and the vanilla client, but this diagram shows the key components and classes within Runelite itself and how they are structurally related



```java
@Test
public void dumpGraph() throws Exception
{
    List<Module> modules = new ArrayList<>();
    modules.add(new GraphvizModule());
    modules.add(new RuneLiteModule(() -> null,  developerMode: true));

    //PluginManager pluginManager = new PluginManager(true, null, null, null, null, null);
    //pluginManager.loadCorePlugins();
    //for (Plugin p : pluginManager.getPlugins())
    //{
    //  modules.add(p);
    //}

    //File file = folder.newFile();
    File file = new File( pathname: "D:\\test\\test");
    try (PrintWriter out = new PrintWriter(file,  csn: "UTF-8"))
    {
        Injector injector = Guice.createInjector(modules);
        GraphvizGrapher grapher = injector.getInstance(GraphvizGrapher.class);
        grapher.setOut(out);
        grapher.setRankdir("TB");
        grapher.graph(injector);
    }
}
```

Figure 11: *Statements that were commented out were those that loaded plugins, which were done to generate a simpler diagram for the architecture.*

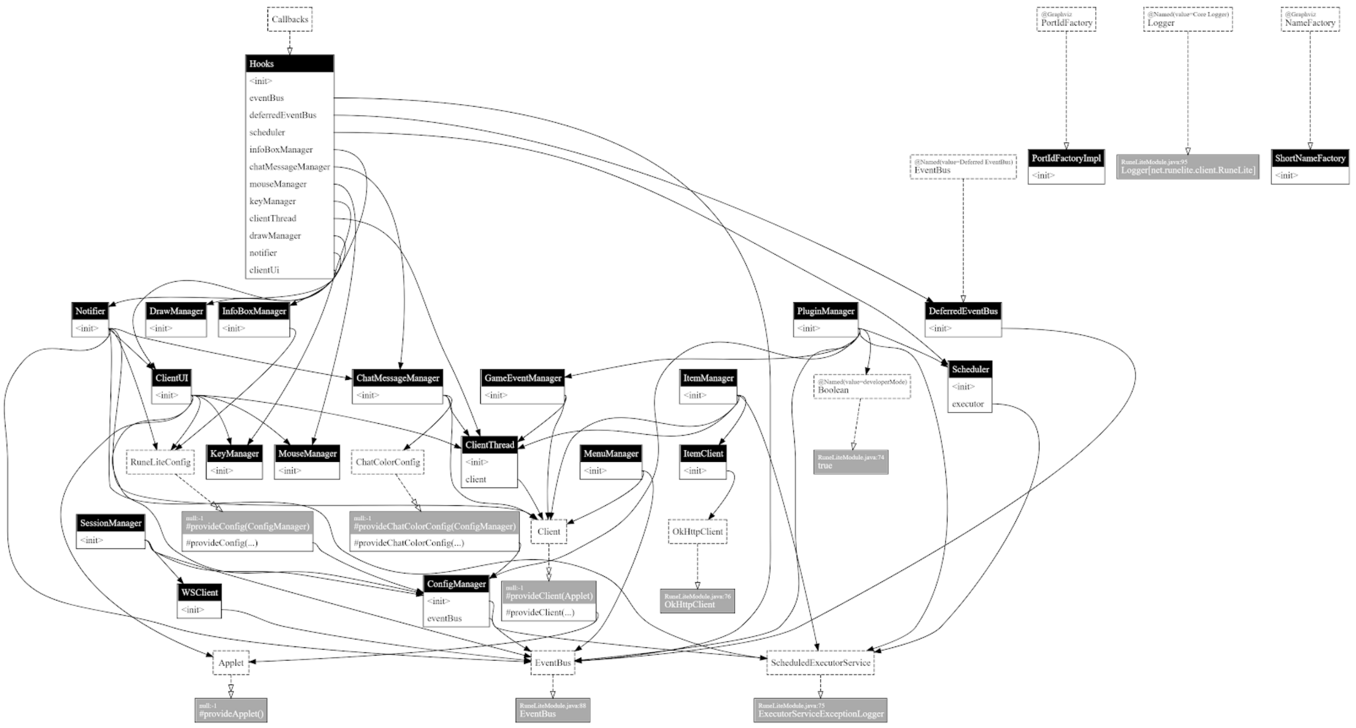
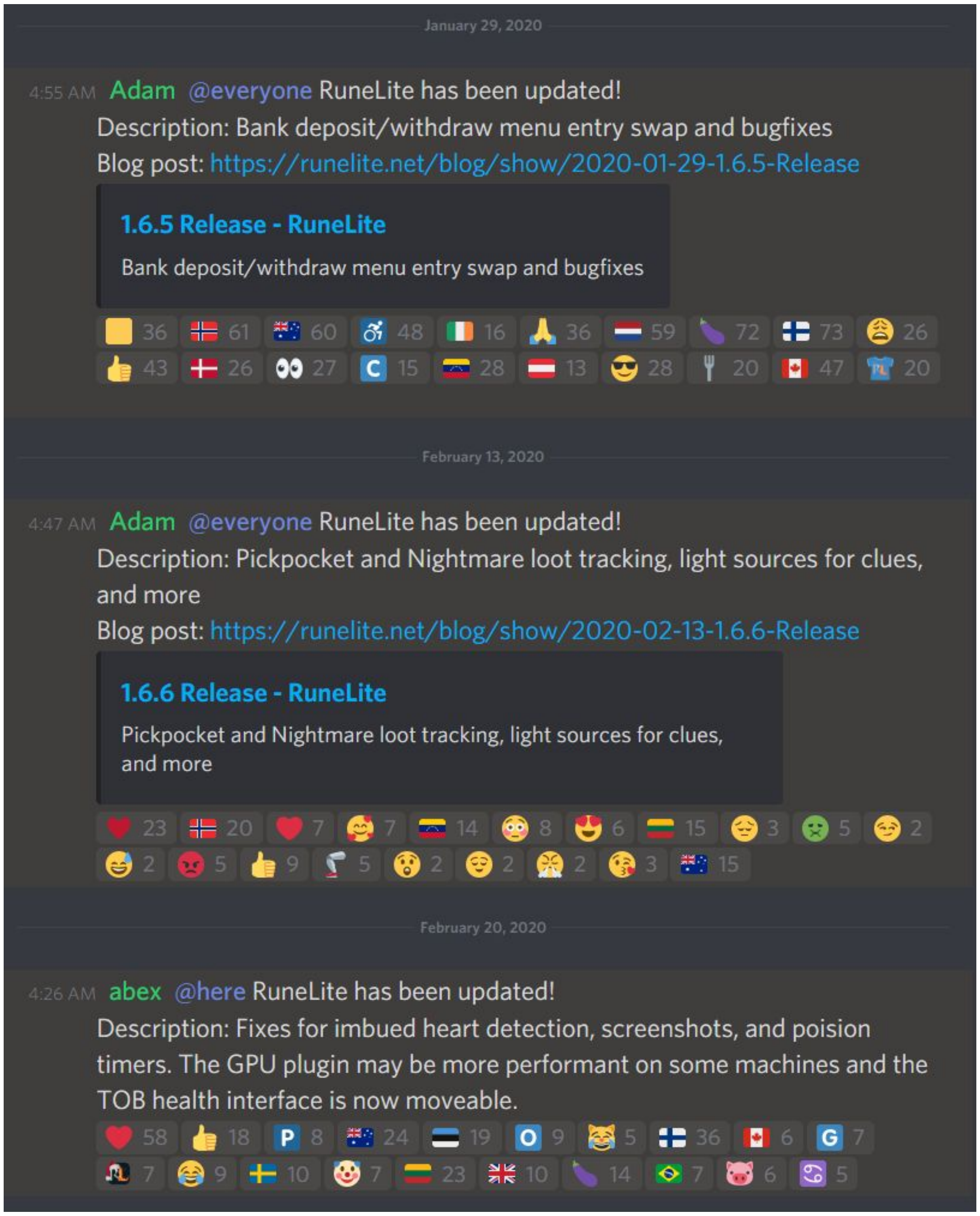


Figure 12: *The resulting diagram without the plugins, and thus the recovered architecture for Runelite*

Social Context

State of the Project

Ongoing development with updates being released every Thursday along with the main game if there is one. See the official discord screenshot below. We can see that with every update, there are release notes detailing what components may have been updated or what bugs were fixed.



January 29, 2020

4:55 AM Adam @everyone RuneLite has been updated!
Description: Bank deposit/withdraw menu entry swap and bugfixes
Blog post: https://runelite.net/blog/show/2020-01-29-1.6.5-Release

1.6.5 Release - RuneLite
Bank deposit/withdraw menu entry swap and bugfixes

February 13, 2020

4:47 AM Adam @everyone RuneLite has been updated!
Description: Pickpocket and Nightmare loot tracking, light sources for clues, and more
Blog post: https://runelite.net/blog/show/2020-02-13-1.6.6-Release

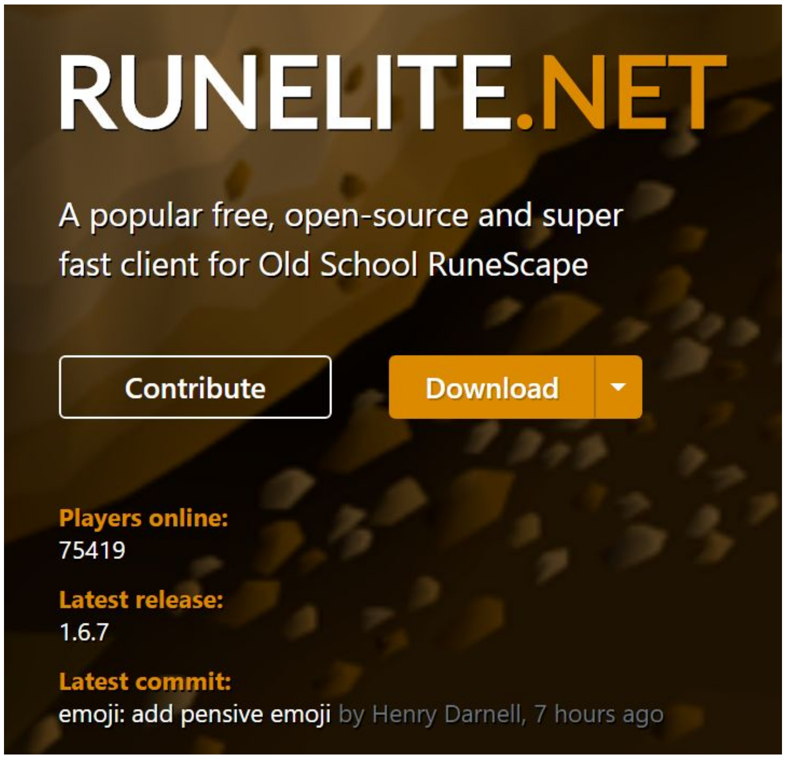
1.6.6 Release - RuneLite
Pickpocket and Nightmare loot tracking, light sources for clues, and more

February 20, 2020

4:26 AM abex @here RuneLite has been updated!
Description: Fixes for imbued heart detection, screenshots, and poision timers. The GPU plugin may be more performant on some machines and the TOB health interface is now moveable.

Level of usage

The two screenshots below are taken at around the same time. Meaning there are more people with the runelite client open than the actual total amount of players logged into the game.



RUNELITE.NET

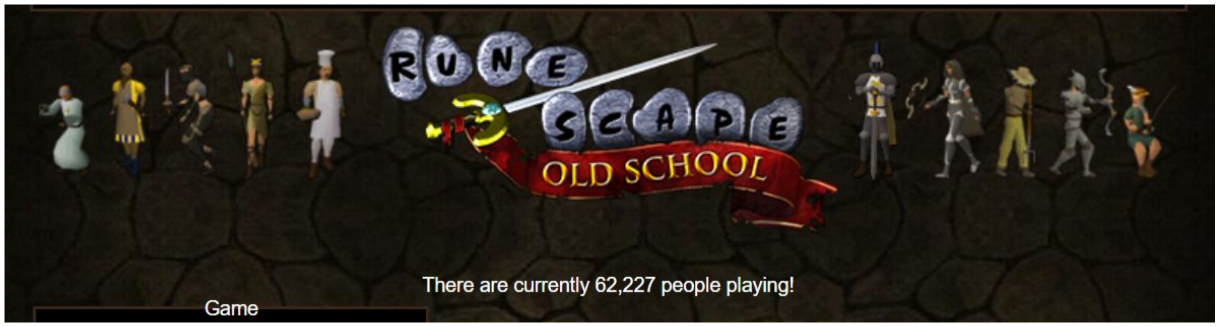
A popular free, open-source and super fast client for Old School RuneScape

Players online:
75419

Latest release:
1.6.7

Latest commit:
emoji: add pensive emoji by Henry Darnell, 7 hours ago

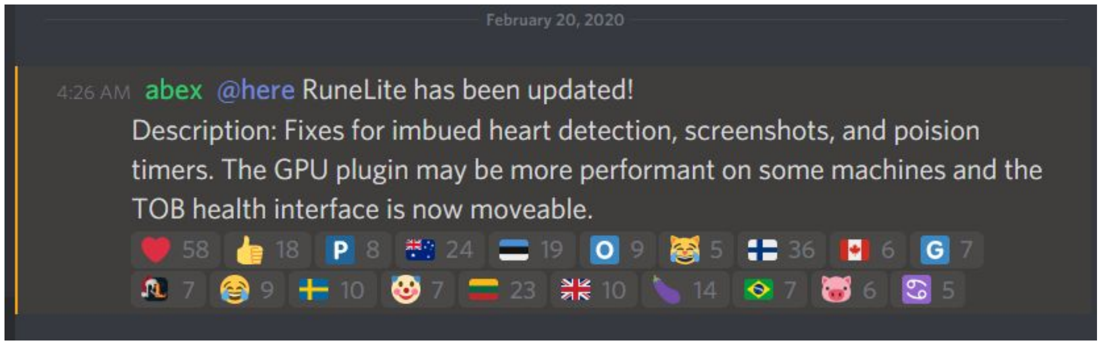
As seen on the left, the number of players online when this screenshot was taken was 75419. This information can be seen on Runelite.net.



There are currently 62,227 people playing!

As seen above, the screenshot shows that there were 62277 people playing Runescape at around the same time that the screenshot for Runelite players were taken.

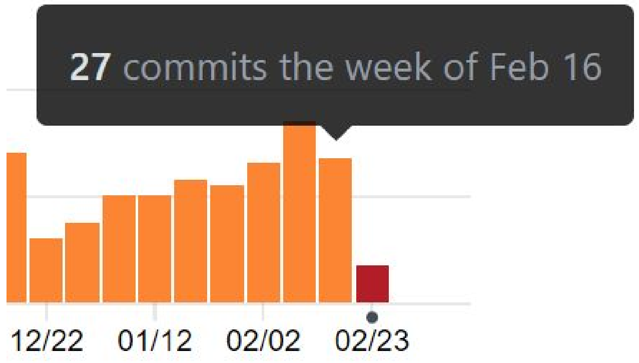
Last release date
Latest release was on February 20th 2020.


February 20, 2020

4:26 AM abex @here RuneLite has been updated!
Description: Fixes for imbued heart detection, screenshots, and poision timers. The GPU plugin may be more performant on some machines and the TOB health interface is now moveable.

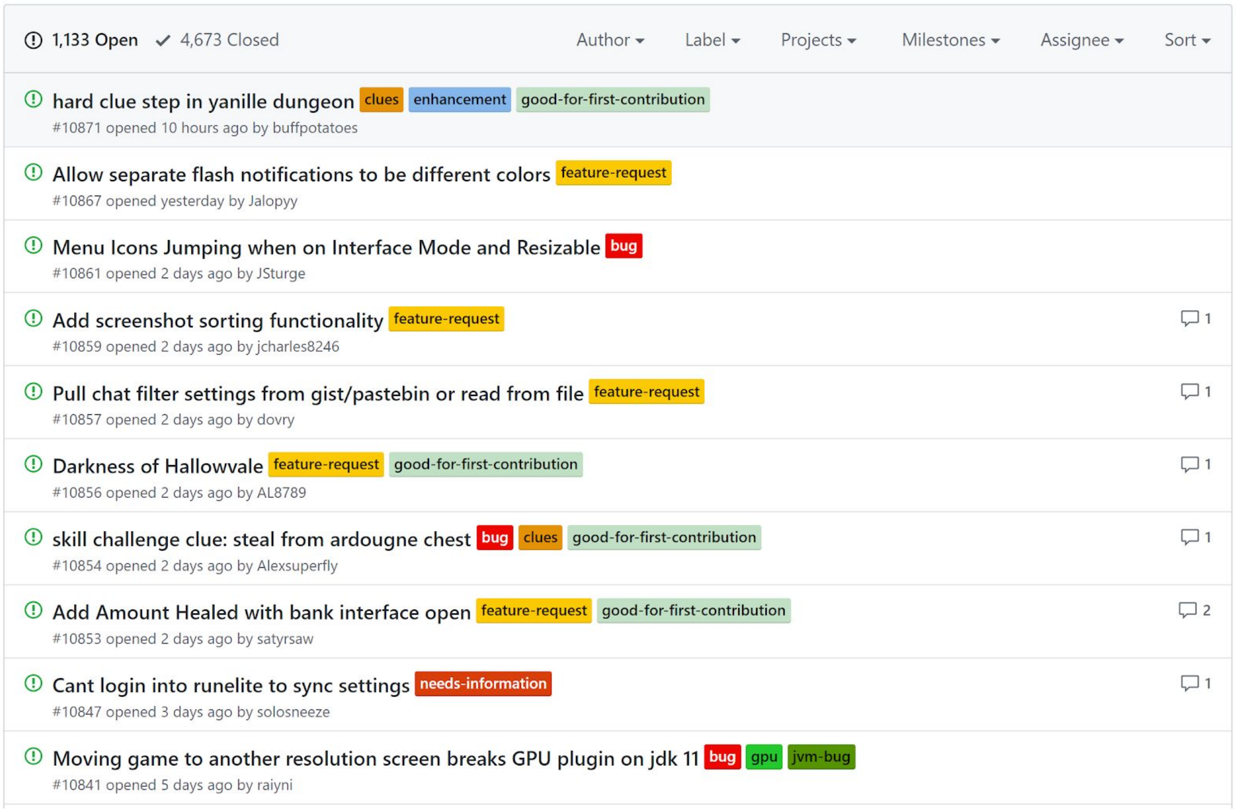
Release roadmap
There is no release roadmap per say, but everything that gets merged into the project usually gets released the following Thursday.

#commits as of late
There have been 27 commits in the last week according to the github insights page.



#open issues as of late

There are 1133 open issues in total, with new ones being added daily by the community of players. This can be seen in the screenshot below. Issues are also being monitored, labeled, and closed actively by the team.



1,133 Open ✓ 4,673 Closed

hard clue step in yanille dungeon clues enhancement good-for-first-contribution
#10871 opened 10 hours ago by buffpotatoes

Allow separate flash notifications to be different colors feature-request
#10867 opened yesterday by Jalopyy

Menu Icons Jumping when on Interface Mode and Resizable bug
#10861 opened 2 days ago by JSturge

Add screenshot sorting functionality feature-request 1
#10859 opened 2 days ago by jcharles8246

Pull chat filter settings from gist/pastebin or read from file feature-request 1
#10857 opened 2 days ago by dovry

Darkness of Hallowvale feature-request good-for-first-contribution 1
#10856 opened 2 days ago by AL8789

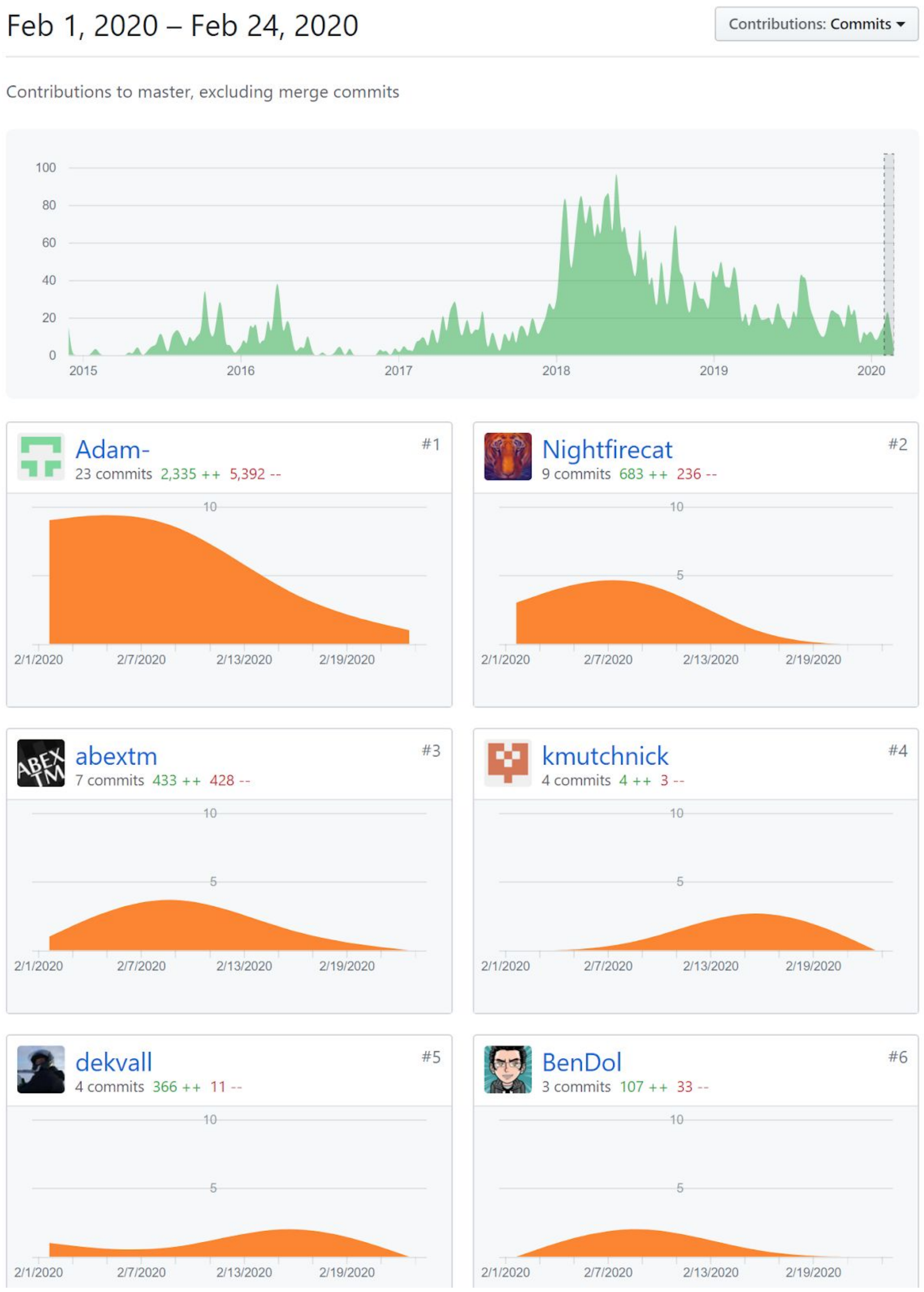
skill challenge clue: steal from ardougne chest bug clues good-for-first-contribution 1
#10854 opened 2 days ago by Alexsuperfly

Add Amount Healed with bank interface open feature-request good-for-first-contribution 2
#10853 opened 2 days ago by satyrsaw

Cant login into runelite to sync settings needs-information 1
#10847 opened 3 days ago by solosneeze

Moving game to another resolution screen breaks GPU plugin on jdk 11 bug gpu jvm-bug
#10841 opened 5 days ago by raiyni

#active developers
If we look at the number of commits just for the month of February 2020 on github, we get the following, and can also see that Adam the creator is still the most active:


Feb 1, 2020 – Feb 24, 2020

Contributions to master, excluding merge commits

Adam- #1
23 commits 2,335 ++ 5,392 --

Nightfirecat #2
9 commits 683 ++ 236 --

abextm #3
7 commits 433 ++ 428 --

kmutchnick #4
4 commits 4 ++ 3 --

dekvall #5
4 commits 366 ++ 11 --

BenDol #6
3 commits 107 ++ 33 --

Standards that we should adhere to
A set of coding convention that should be followed detailed here:
https://github.com/runelite/runelite/wiki/Code-Conventions

Standards include:

1) Java conventions but use tabs instead of spaces
2) Braces should be placed on the next line
3) A header detailing the copyright should be placed in each file
4) Imports in one logical block, blank space to separate logical blocks, and extra tabs for multi-line annotations

The code conventions for Runelite also suggests that we use IntelliJ's formatter or install a Checkstyle-IDEA plugin that will check for styling standards as we type.

Typical process for contributing
Runelite has a set of contribution guidelines posted here:
https://github.com/runelite/runelite/blob/master/.github/CONTRIBUTING.md

They direct us to their discord server if we have any questions, and to submit an issue if we have found a bug. They also advise that we avoid submitting duplicate issues. The Runelite team suggests that we provide the following information before submitting an issue:

- Overview of the issue
- Java Version and Operating System
- How to reproduce the error
- If any related issues have been reported
- Either fixing the bug ourself or pinpoint where in the code we think the fix should be made

The typical process for submitting a pull request would be to first fork and clone the repo, then set a new remote upstream for syncing, then make changes to a separate branch while testing and following the code conventions, and finally commit and push to Github before sending a pull request. The changes will be reviewed by the team and they will decide whether we have to make changes again before they can merge it to the main repo.

Tools that support the process
Bug report template
https://github.com/runelite/runelite/blob/master/.github/ISSUE_TEMPLATE/Bug_report.md

1. Check if someone else already noted the issue
2. Describe the bug
3. Provide clear steps to reproduce bug
4. Describe the expected behaviour

5. Add screenshots, other attachments, and any contextual details like the operating system

Feature request template
https://github.com/runelite/runelite/blob/master/.github/ISSUE_TEMPLATE/Feature_request.md

1. Describe feature request and any related problems
2. Describe solutions
3. Describe alternatives
4. Provide any additional context

Client developer tools
https://github.com/runelite/runelite/wiki/Using-the-client-developer-tools
To enable this tool in IntelliJ, we would have to edit the configuration for our run, and under program arguments, supply it with --developer-mode and also --debug. We would also have to supply the VM arguments with -ea.

Runelite also uses Travis for testing purposes, git/github for version control and tracking issues, and Maven for project management and build.

Interesting pull requests:
https://github.com/runelite/runelite/pull/10664 - This request adds two features: a configuration for the Ctrl+F11 hotkey which hides the sidebar entirely and a configuration for the hotkey Ctrl+F12, which by default toggles the current or last plugin panel.

https://github.com/runelite/runelite/pull/7836 - Very detailed pull request that is still [WIP] with many screenshots and commits. This feature has garnered the support of many players, and many have even manually merged the pull request into their local repo for testing. Might also be very popular because it is related to one of the most popular skills in OSRS, Slayer.

https://github.com/runelite/runelite/pull/2821 - This is one of the oldest still open pull request for Runelite, everything is approved and ready to merge, but the devs realize that there is probably a better way to do it, and never merged the pull request. What's interesting is how nobody actually got around to adding text wrap the better way, and so this pull request stays open in limbo.

https://github.com/runelite/runelite/pull/10683 - This one is interesting because the person had an issue with a plugin missing a 1 click teleport option for a specific in game item, implements the feature, submits his pull request, and then realizes half and hour later that his implementation will be problematic and cause confusion for players, so he comments on his pull request his reasoning and closes it himself. This teaches us to think first before doing, because he probably wasted a lot of time implementing a feature that he himself didn't even like in the end.

https://github.com/runelite/runelite/pull/10815 - This is a pull request that was shutdown by the reviewer. The person has also made several pull requests immediately prior to this one, but decided to merge them all into one. He obviously didn't read any of the contribution guidelines, and his solution to the problem was also lacking. This is interesting because ift gives us an example of what not to do when contributing.

Interesting Issues:
https://github.com/runelite/runelite/issues/10704 - Add search bar in loot tracker plugin, most players like to leave the loot tracker there to continuously track loot from all drops in game. After awhile it becomes very cluttered and hard to find specific things. A search bar will help with finding specific monsters, or even specific items and all monsters that have dropped that item.

https://github.com/runelite/runelite/issues/10416 - Very good idea actually. The client has a sidebar with many useful functions and this request wants to separate the sidebar and the rest of the client. Might even be a good idea to have all the different items in the sidebar all able to pop out into their own windows. Makes tracking different things easier with the second monitor.

https://github.com/runelite/runelite/issues/10015 - This requests an option to change the chatbox's opacity for easier viewing. Currently when in awkwardly lit areas of the game, chat will be barely readable, having a slightly opaque chat background will help immensely.

https://github.com/runelite/runelite/issues/6410 - Much needed feature, when you die in game, you have a 60 minute window to pick up all your lost items from the ground where you died. Runelite gives you an icon with a 60 minute timer to count this down for you, but blocks an area of screen space. When the timer is no longer needed, it is currently a huge hassle to disable the icon. A simple right click clear option on the icon will be much more intuitive than the current implementation of going through the list of plugins and disabling/re-enabling it. Other overlay plugins already have similar right click clear functionality, and it is surprising how something so simple has not been implemented already. With the issue being on Github for almost 2 years now.

https://github.com/runelite/runelite/issues/9601 -This is one of the 3 open issues with a "help wanted" label, we find it interesting because this is an issue that can only be solved by the community pooling resources together and gathering data from the game. Deon has also contributed to this issue multiple times via comments with in-game screenshots.