Graphkasten

Kryštof Albrecht
krystofalbrechtus@gmail.com

March 14, 2021

Contents

1 Introduction

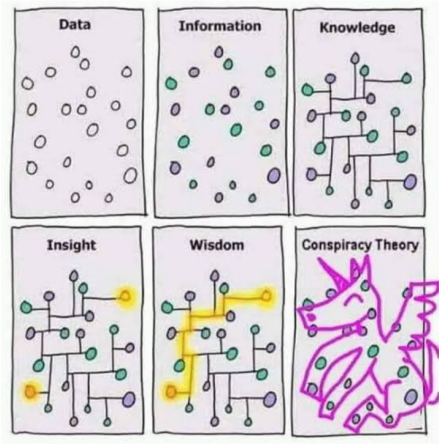
Graphkasten provides a *graph view* of the notes, *keyboard-driven navigation* and *flexible search* to refine interaction with a Vimwiki-based Zettelkasten.

1.1 Motivation

Tools like Vimwiki, Vimzettel and Taskwiki provide a solid base for a Zettelkasten workflow. These are based on *the Vim text editor*, which has a lot of advantages:

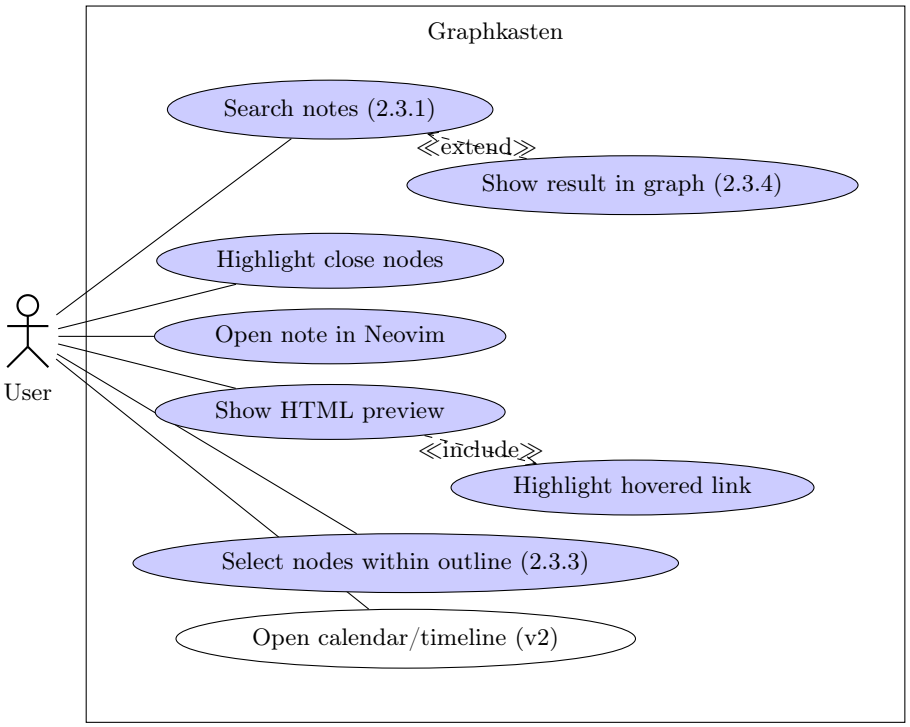
- Text editing cannot be faster
- Adding new notes is effortless
- The system is simple
- Tasks can have note links in them
- Any missing functionality can be added through Vimscript

The problem is however, that there is no GUI. Tools like *Obsidian* offer a *graph view* that enables natural viewing of the *networked structure* of the Zettelkasten. This view allows faster and easier search and navigation, but also improves the chances of *emergent ideas.*



1.2 Use case diagram

The note[1] graph is *the basis* of the system and *all* of the use cases are related to it.



1.3 Miscellaneous features

Apart from the aforementioned main use cases, there will also be the following features:

- Showing files/attachments in the graph view
- Support for Vimwiki tags
- Real-time updates
- Fast search

[1] The terms *node* and *note* are used interchangeably throughout this document.

1.4 Common scenarios

Here are a few scenarios of how using the application might play out.

1.4.1 Processing the staging area

I've taken some notes and put them into the Staging Area. Now I want to quickly integrate them into the Zettelkasten.

1. The staged notes have an *outline* around them in the graph view. Each staging subtask is *further divided* by outlines.
2. I *highlight* all the task notes within an outline.
3. I *open* all the notes at once in Neovim with a single click.
4. I can view the HTML preview of the note I am working on.
5. As I work, the graph updates *in real time*.

1.4.2 Searching for information

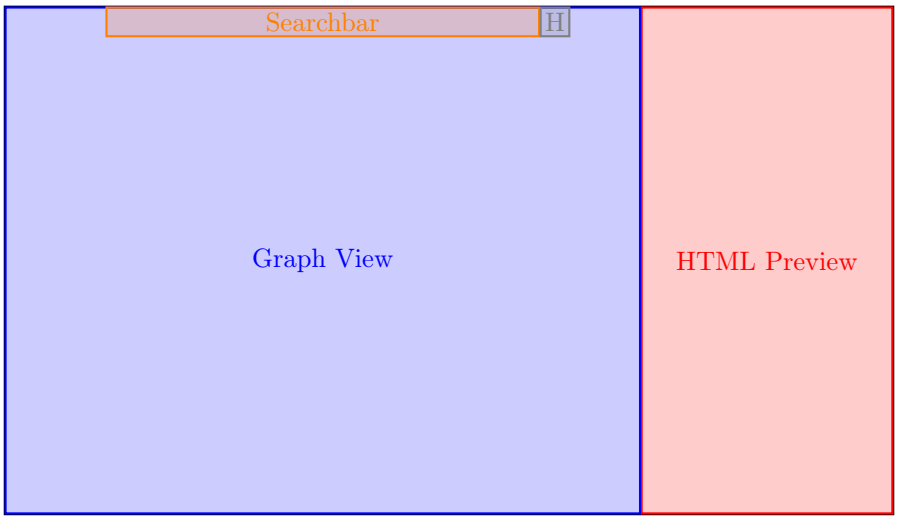
I need to query some information in the Zettelkasten.

1. I enter my search query by *just starting to type*.
2. The graph updates in real time and shows just the nodes matching the query.
3. I have the option to show a small snippet of the note next to each node.
4. The most relevant node is highlighted and its HTML preview is shown automatically.
5. Hitting ENTER, I can navigate the nodes with *Vim keys*.
6. Hitting ENTER again, I can navigate the HTML preview.

2 Architecture

2.1 UI Overview

The note graph is *the basis* of the system and so will always be shown. The user will then be able to *adjust* this graph view. Any other view will open as a sidebar.



The *graph view* is the most important part of the UI. It visualizes stored ideas and the connections between them and allows navigating related ideas easily.

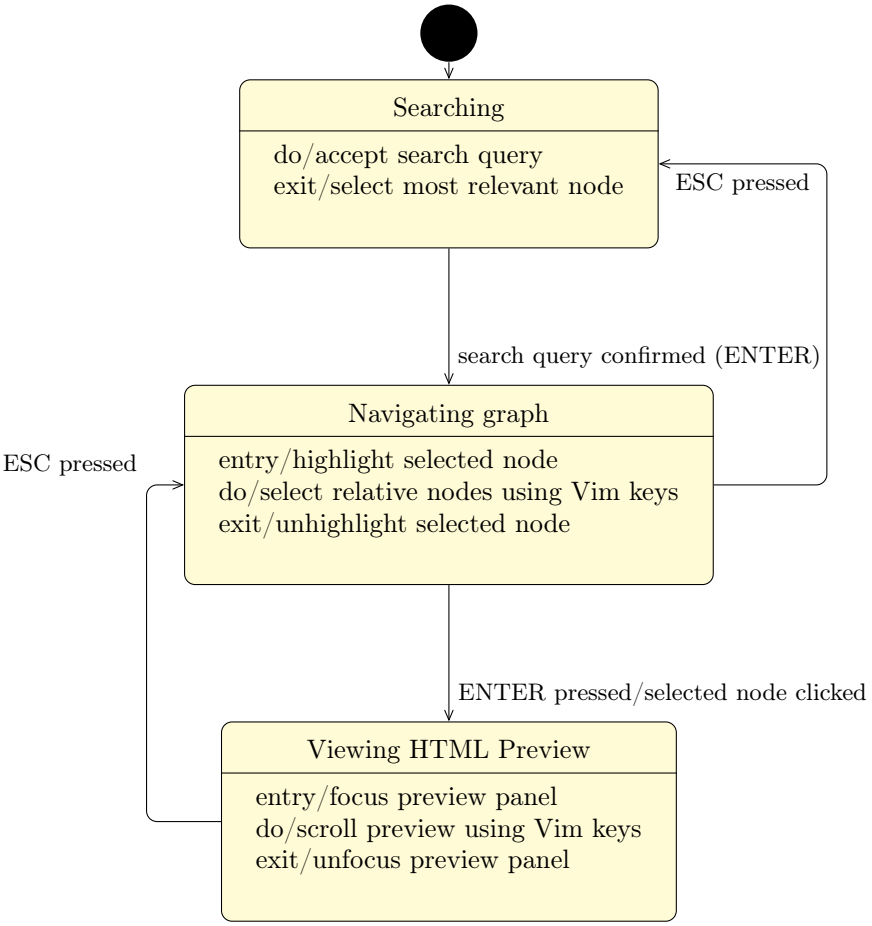
The *searchbar* allows filtering what is shown in the graph view. Nodes matching the search will be shown and the view will zoom in to fit the results. Other nodes will be hidden.

The *HTML preview* shows the currently selected note in HTML format. Search results are also highlighted on the page. Clicking links to other nodes on the page selects that node in the graph.

The *hamburger menu* shows configuration options.

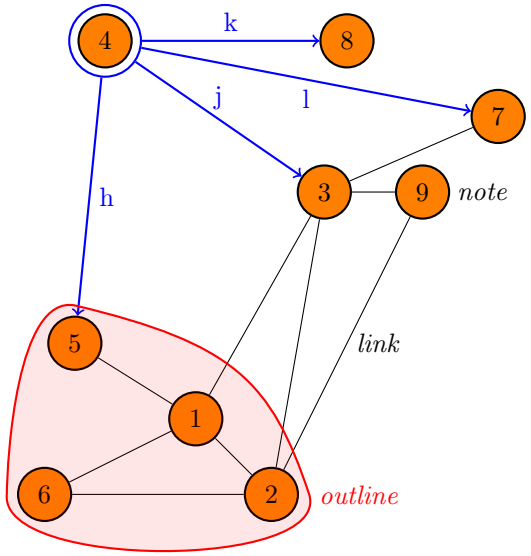
2.2 System states

This section describes the states the system can get into through user interaction. The application can be controlled using the mouse, but also allows for full keyboard control.



2.3 Graph View

This section provides an overview of the various *elements* of the graph view.



2.3.1 Notes

2.3.2 Links

2.3.3 Outlines

2.3.4 Lenses

2.4 Search

2.5 HTML Preview

2.6 Options