Version 2.0

User's Guide & Technical Overview

Revised April 3, 2014

Table of Contents

Evernote and NixNote Overview

Evernote is company which provides note synchronization tools across multiple platforms. Evernote provies clients for Windows, OS-X, iOS, Chrome OS, Android, Windows Phone, BlackBerry, and web based clients. Their goal is to be a “remember everything” type of service which allows you to search text, handwritten, images, and attachments regardless of where you are or what platform you are using. More information can be found at http://www.evernote.com.

Absent from their support is Linux. When asked about Linux their usual response is that the market is too small to warrant a client and Linux users can use the web interface to access their notes.

NixNote is designed to use Evernote's public API to provide a native full synchronization client for Linux users.

NixNotte's features include:

Full synchronization of all notes & attachments. This includes linked & public notebooks as well as private notebooks.

The ability to create, edit, and delete notes, tags, notebooks, and saved searches.

The ability to search notes and the ability to index some attachments.

The ability to use the image text recognition features that Evernote provides.

Support for multiple Evernote accounts.

NixNote History

This project began several years ago as a side project written in Java. The initial goal was not to provide a full client, but to allow me to learn more about Evernote and to attempt to fulfill a few personal needs. Eventually it grew into a full client and was released under the name “NeverNote” in the hopes that others found it useful.

NeverNote was eventually renamed to “NixNote” at Evernote's request because they felt the name was too similar to Evernote and didn't want to confuse their users.

NixNote went through several releases until it reached 1.4. At that time, I decided to begin a rewrite to bypass some of the issues I had encountered with the Java and my . The overall goal of the new NixNote 2 was to provide a better user experience while still providing as many of the same features as NixNote 1.x. There will most likely not be any new releases of the Java based NixNote 1.x beyond version 1.6.

Differences between Evernote and NixNote

I've tired to keep NixNote with the same feature set as Evernote. Despite this, there are some differences. Some of these I hope to correct as time allows, some I'll never be able to correct due to limitations, and some I may never correct unless someone asks since I don't use them..

Features in Evernote that NixNote doesn't have or does differently:

- The search syntax is slightly different. I think all the same search features work, but NixNote allows any term to be negated, where Evernote does not. For example, searching for the term “notebook:inbox” will find any note in the inbox notebook, but searching “-notebook:inbox” will search for any note containing the word inbox and doesn't contain the word “notebook”. NixNote interprets this differently. It will search for any note that is not in the notebook inbox.
- Twitter & Facebook sharing. I don't use it so I don't plan on implementing it. If you need it let me know and I'll consider it, but it isn't high on the list of priorities.
- Audio notes are not directly supported. Evernote allows you to directly record a message, but NixNote does not. The best workaround is to create the audio file in another application and attach the file to a note.
- Ink notes are not possible. Evernote doesn't provide an API for ink notes and reverse engineering the file format is something I can't do. You can view ink notes as an image, but that is it. Many of Evernotes non-Windows clients have this same restriction.

There are a few small features that NixNote has that Evernote doesn't. None of them are earth shatteringly cool features, but I like them.

- You can close or open notebooks to make them disappear. This is similar to the selective sync feature, but it is different. It allows you to do full synchronization of a notebook, but you can hide notebooks you don't frequently use. I use this feature for some older reference material that I seldom need, but I don't want to forget. This just hides them in the interface so when I search I don't need to look through all those old notes. The important thing to remember is that this does NOT impact any synchronization of a note.
- Business acounts are not supported. They may work, but I've never tried them.

- Totals for tags & notes are dynamically updated filtering notes. For example, if you select the notebook “inbox” the counts beside the tags change to only be the count of notes matching your selection criteria.
- Editing the source of a note is possible, so if you REALLY want to mess with the formatting or just want to see what it looks like you can do an Edit/View Source. I don't recommend it for everyone since you can really mess things up, but it can be useful at times.
- Notes can be “pinned”. A pinned note will always appear in the note list regardless of any other selection criteria. If a notebook is closed, however, a pinned note will not be visible.
- You can “quick link” from a note to another note by highlighting text & selecting “Quick Link” from the context menu. It will then search for any notes with titles matching the selected text.
- LaTeX formulas can be inserted as PNG images and edited later when needed. You must have the program “mimetex” installed for this to work.
- HTML entities can be entered directly via a dialog box.

Installation

NixNote can be installed in various ways depending upon the distribution you are using.

Debian based distributions (including Ubuntu) can use the deb packages. Once downloaded use the command “sudo dpkg -i <filename>” and the deb will be installed.

RedHat based systems cad download the RPM. Once downloaded use the command “sudo rpm -uvh <filename>” and the RPM will be installed.

For both RPM & DEB packages, you will most likely need to download and install any dependencies your system is missing.

If you are using a distribution other than a RedHat or Debian based system you can download the tar file. Once downloaded and unpackaged you can install it by running the “sudo ./install.sh” script from the directory. There is no automatic checking of dependencies with a tar file, so you will need to validate the dependencies manually.

As a final resort, you can download the source and compile it yourself. The build instructions are in the development section later in this document.

Once installed, NixNote should appear under the “Networking” section of your menus. The first time you run NixNote, it will create a database in ~/.nixnote/ directory.

Interacting with Evernote

Initial Synchronization

NixNote is designed to interact with Evernote and synchronize any data you have. The first time you synchronize it will download all of your data. The first sync can take a very long time based upon the size of your database.

If it encounters problems and crashes on the first sync, you should be able to restart the program and continue the synchronization. It will take up the synchronization where it stopped.

Storing your userid & password

NixNote does not have access to your userid & password. Authorization is granted through Evernote directly when you first synchronize. This gives NixNote an authorization token that is good for one year. When the year expires you will be promted for another userid & password. If you desire, you can revoke NixNote's access at any point via your account on http://www.evernote.com.

Synchronization intervals

NixNote will synchronize every 15 minutes by default. You can customize this under the Edit/Preferences dialog in the “Sync” section.

There are a few other options you can customize if you wish.

- Sync Notifications – This provides a visual message from the system tray when a synchronization is complete. You must have the tray icon enabled (unde the Appearances) tab for this to be enabled.
- Sync on Startup – This option will automatically log you into Evernote and start a sync when you start the program. If you haven't granted NixNote account access via the login page, this option won't work.
- Sync On Shutdown – By choosing this option NixNote will do a final synchronization when it shuts down. This option will only work if you have granted NixNote access via the login page at least once. Depending upon the amount of data, this could cause NixNote to run for a while in the background.

Emptying the trash

When you delete a note it is moved to the trash. The next time you synchronize that note should then be moved to the trash in your Evernote account. This works the same as any other Evernote client.

There is I difference in behavior between NixNote and the way Evernote's official clients handle the trash. When you empty the trash on an Evernote client it is permanently removed from your local database and when you synchronize it is permanently removed from your online account.

NixNote doesn't work this way. In NixNote, when you empty the trash you can no longer restore it and the note vanishes from your trash, but when you synchronize it will remain in your trash on in your online account and is not permanently deleted. If you empty the trash on another Evernote client, the note is then permanently removed from NixNote. If you restore the note from Evernote's trash it will reappear in NixNote.

This is due to an Evernote security restriction. In the past, they had third party API developers delete the wrong notes and cause work for their support team to restore the note. To prevent this they don't allow other clients to permanently delete notes.

Using multiple Evernote accounts

NixNote will allow you to access multiple Evernote accounts at one time, but these accounts are stored in different databases. To do this , look under the "File" menu option and click "Add Another User". This will give you a dialog where you can specify the name of this account and (if desired) allow you to change to a different Evernote service. After creating a new account, you can switch to that account and do a sync. Authorization is not shared between accounts, so you can have multiple

Using NixNote

Notes

Ink notes

NixNote allows you to create a text note and synchronize it with your Evernote account. You can't create or edit ink notes. Evernote doesn't provide an API or any documentation on the file formats for ink notes so there is nothing I can do but vewing ink notes is possible. When you synchronize it should pull an image down and show it to you. The note is read-only and you can't change the contents.

Evernote's non-Windows clients cannot edit ink notes, so I'm not expecting Evernote to provide this ability to other third party clients any time soon.

Merging notes

You can merge the contents of multiple notes together. This is done by selecting multiple notes in the note list and choosing "Merge Notes". The order of the merged notes is the same order in which you selected them.

Note history

If you are a premium user, you can restore older versions of notes from Evernote's servers. This is under the "View" menu option "Note History".

Duplicating a note

NixNote allows you to duplicate an existing note. Any data or file attachments are treated as separate notes. This can be useful if you wish to use one note as a template for other notes.

Background color

You can change the background of any note by right clicking in the note and choosing the "Background Color" option. This is a non-standard feature and it may not work on all Evernote platforms. Depending upon your web browser it may appear properly in the web interface, but editing it in the web interface will remove the color.

Viewing note lists

You can customize the list of notes and you can customize where it appears. By default it is on the right hand side of the screen just above the note editor. By selecting "View/Narrow List View" or "View/Wide List View" you can move it from appearing above the note to appearing between the note and the left hand panels.

If you don't wish to change which columns are visible in the note list, you can customize them by right clicking the title bar and choosing the columns to view. The "wide view" and "narrow view" are separate so customizing one does not

update the other. This is to allow for things like a thumbnail in the narrow view where you have more vertical space, but hiding it in the wide view where you want to be able to see a larger number of notes.

Printing

Printing under Linux requires that CUPS be installed.

Note HTML Source

You can view the HTML source of a note under the View/Show Source menu option. Doing this allows you to change the markup of a note more directly and overcome some of the editor's limitations, but it also means you can really mess things up. NixNote won't prevent you from producing really badly formatted HTML but it will still try to clean up anything you create so it can synchronize with Evernote properly.

Note Links

Note links can be created in NixNote by right clicking on a note in the note list and selecting “Copy as URL”. You can then paste the URL into another note. When you click on the link, the note will open in a new window.

Notebooks

Creating, editing, & deleting notebooks

Notebooks can be created or renamed the same as any Evernote client. When you create a notebook you have the option of creating it as a local or synchronized notebook. Local notebooks are saved only on your hard drive and never go through Evernote's servers. If you local notebooks I highly recommend doing frequent backups to preserve any data in the event of a crash.

You can move notes from synchronized notebooks to local notebooks. If you move a note to a synchronized notebook it will be created on Evernote the next time you synchronize. If you move from a synchronized notebook to a local notebook that note is moved to the trash on your Evernote account the next time you synchronize.

Renaming a notebook works the same as any other Evernote client. The next time you synchronize it will update the name on Evernote. NixNote will not permit two notebooks to have the same name and it is case insensitive, so you can't have a notebook called "My Data" and another called "my data". NixNote considers them to be the same. Evernote's clients work this way too.

Deleting a notebook is possible only if all notes in that notebook have been deleted or moved to other notebooks. It doesn't allow you to delete a notebook and all notes in one operation.

Stacks

NixNote supports stacks the same as other Evernote clients. To set a stack right click on the notebook and select "Set Stack" or choose the option "File/Notebook/Set Stack". There, you can set a stack name or choose an existing stack for that notebook. To remove a notebook from a stack simply blank out the stack name and click "OK". It will then be removed from that stack.

Opening & closing notebooks

NixNote has a non-standard feature to permit you to show or hide specific notebooks. This allows you to, for instance, hide all your personal data on your work PC or hide notebooks you seldom use.

To open or close a notebook, choose the option "File" menu under"Open/Close Notebooks". A dialog box will appear where you can choose the notebooks you wish to view.

Please note, this does NOT change any synchronization behavior. Any changed notes are synchronized regardless if the notebook is opened or closed and all data remains on your hard disk. If you don't wish to have this feature, then you need to do a selective synchronize.

Tags

Tags in NixNote work in much the same way as other Evernote clients. You can't have duplicate tag names and selecting a parent tag does not show notes containing the children of those tags. This is to mimic Evernote's interface.

When you create a tag it is placed in the top level of the tree. If you want it somewhere else you need to drag it to that tag and it will be moved to be a child of that parent tag.

Merging Tags

You can merge multiple tags into one tag. To do this, select the tags you want merged (including the one you want them merged with). Right-click in the tag menu and select the "Merge" menu option. This will bring up a dialog box of the selected tags. Choose the tag you want them merged into and click OK. Any notes with any of the selected tags will be changed to the one tag. Please note that the old tags are not deleted.

Saved Searches

Saved searches work the same as in the Evernote client. I believe the full syntax is supported, but as Evernote announces new search keywords NixNote may be a little behind implementing them.

The current items supported are:

- Words in the title of a note (intitle:)
- Tags (tag:)
- Notebook (notebook:)
- Author (author:)
- Source (source:)
- Source Application (sourceapplication:)
- Created (created:)
- Updated (updated:)
- Subject Date (subjectdate:)
- Todo (todo:)
- Recognition Type (recotype:)
- Stack (stack:)
- Longitude (longitude:)
- Latitude (latitude:)
- Altitude (altitude:)

In addition to the search terms, you can prefix with a term with a negative (i.e. -tag:) and it will negate the statement. Using this, you could search for all notes that don't match the criteria (for example, all notes not tagged with a value).

Deleting Notes

Deleting a note moves it to the trash. It can be restored at any time until the trash is emptied. When the trash is emptied, the note is not permanently deleted from NixNote until you empty the trash on an Evernote client. The section “Emptying the trash” above has more details.

Screen Clipper

NixNote does have a screen clipper, but it may not work on all systems. To use it, go to Edit/Preferences/“Appearance” tab and check “Show tray icon”. Once you see the tray icon, you can right click to bring up a menu and choose “Screen Capture”.

If the screen clipper doesn't work for you or you have a favorite screen clipper, you can mimic this behavior by setting up an import folder (look under Tools/Import Folders) and save your clipped images to that directory.

Colored Notes

NixNote allows you to change the background color of a note. This isn't officially supported by Evernote, but it will display properly on some of their clients. You can set the background color by right clicking in the note and selecting “Background Color”. Be aware that editing the note on an Evernote client or in the web version of Evernote may remove this color. It depends upon the client.

Keyboard Shortcuts

Most of the menu items in NixNote can be customized. To do this, you must create a shortcut.txt file in your ~/.nixnote directory. For more information on customizing shortcuts, look in /usr/share/nixnote2/shortcuts_howto.txt. After making any changes, you must restart NixNote for the changes to take effect.

Preferences Dialog

Under Edit/Preferences there are various options that can change the appearance and behavior of NixNote.

The first tab you will see is the Appearance tag.

- “Show tray icon” - If you window manager supports it, this will show an icon in the tray of your status bar.
- “Show splash screen on startup” - When starting NixNote, display a splash screen until the startup is complete.
- “Display PDFs Inline” controls how NixNote displays notes with PDF attachments. If selected, it will try to create a PNG image of the first page of the PDF. The user can then use the arrow keys above the image to browse through the PDF without leaving NixNote. If this value is uchecked, the PDF will be displayed as a small icon which, when clicked, will launch the default PDF reader.
- “Display missed reminders on startup” - If NixNote wasn't running when a reminder was due, this option will show it when NixNote starts the next time. If unchecked, you will not receive any notifications of missed reminders.
- “Always start minimized” - Always start NixNote as a minimized application. This is particularly useful if you have it set to autostart and minimize to the tray.
- “Start automatically at login” - Starts NixNote whenever you login.
- “Startup Behavior” - This controls what notes you see when NixNote starts.
    - “Restore Selection Criteria” - Whatever search & selection criteria you had when NixNote shut down the last time will be restored.
    - “Select Default Notebook” - Always start NixNote and just view any notes in the default notebook.
    - “View All Notebooks” – View everything when NixNote starts
- “Default Editor Font & Size” - These allow you to specify the default font & font size used when viewing notes.
- “Window Icon” - Different window icons look good with different window managers & themes. This allows you to change the default icon. On some systems, this may take a restart before it becomes active. On Unity, the

default icon may stay around until the next reboot (this is because Unity caches some images).

The next tab is the “Locale” tab. It is used to customize how dates & times are displayed.

The Indexing tab is discussed in the “Indexing” section of this document.

The “Sync” tab discussed in the "Synchronization” section above.

Next is the “Search” tab. This tab controls how NixNote resolves searches.

- “Index Attachments” - If checked, NixNote will use LibreOffice to turn various attachments to text files, which are then indexed by NixNote. This can dramatically increase the size of your database and can be CPU intensive when the index is happening. If you do not have LibreOffice installed, this option has no effect. When this option is changed, it only impacts new attachments that have not yet been indexed.

- “Minimum Recognition Weight” - When Evernote does text recognition of images, it returns a list of possible words in the image along with a weight. The higher the weight, the more certain Evernote is of a match. This value is used to determine the minimum recognition certainty for NixNote to consider it a match when searching. Changinngng this value does not require you to reindex your database.

The “Debug” setting controls various debugging settings.

- “Disable uploads to server” - When checked, notes will only be downloaded and no changes done locally will be sent to Evernote. This is useful if you wish to test Evernote without running the risk of corrupting your notes.

- “Show LID column” - This adds an additional column to the note list that showes the LID (local ID) for each note. This is primarily used for debugging and most people really don't care about it.

- “Message Level” - Specifies the amount & type of messages that are written to the log file. Normally, this should be set to “Info”, but changing it can be helpful when problems occur.

    - “Trace” - Produces a lot of messages about the various routines that NixNote is calling internally. It is useful in identifying potential bottlenecks. This message level shows all other messages as well as trace information.

- “Debug” - Show all debugging messages. Debugging also shows informational, warning, & error messages.
- “Warning” - Show only warnings & critical errors.
- “Error” - Only ever show critical error messages.
- “Fatal” - Show only the most critical messages that cause NixNote to fail.
- “Info” - Show any informational or error messages. This is the normal message level.

Indexing

Interval and synchronization

Indexing is done via a background process. This process starts periodically and scans for any notes which need to be indexed. This means that a note index might not be updated for a few minutes after the note was updated.

If you wish to temporarily disable indexing, you can stop it by selecting the "Tools/Disable Note Indexing" option. Selecting this option a second time will re-enable indexing and it will automatically be re-enabled when the program is restarted

If LibreOffice or OpenOffice is installed, and the program "soffice" is in the $PATH, NixNote will index MicroSoft Office, Libre/OpenOffice, & PDF documents it finds.

Reindexing specific notes or the entire database

Indexing your entire database takes time, but if you want to do it again you can tell NixNote to do this by choosing the "Tools/Reindex Database" option.

When the entire database re-indexed the search words for that note are not immediately deleted, but are deleted just prior to indexing that note.

You can determine how many notes need to be indexed by looking at the "Tools/Database Status" menu option.

OCR Data

When Evernote gets a note, it will OCR any images in a note and (if you are a premium member) it will OCR any images in PDFs. Evernote stores the OCR text and assigns a value to it. The higher the value the more certain it is that it found a word. The next time NixNote syncs, Evernote will send that OCR data and NixNote can then add it to its search database.

Data which has not passed through Evernote's servers (local notebooks and unsynchronized notes) does not contain any OCR data.

Attachments

NixNote has the ability to index PDF, Microsoft Office (Excel, Outlook messages, Word, & PowerPoint), , ODF, PDF, and text documents. This isn't perfect but in my limited testing it seems to work reasonably well. This does, however, increase the database size immensely if you have a lot of attachments. You can enable or disable this under "Edit/Preferences" in the "Indexing" section. If you enable this option it will NOT reindex any existing attachments, so you may need to reindex your entire database to get all of your attachments indexed.

Editing Attachments

By clicking on a file attachment the default program for that file type is launched. If that program allows editing any changes made will be updated in NixNote. For example, if you open an image in a note using an external editor, any changes to that image will be reflected in the note. This works the same as Evernote's clients.

Tools

Import folders

Import folders are used by NixNote to automatically import data when it appears in a directory. There are two types of import folders.

The first type is “Import and keep”. This type of directory is monitored while NixNote is running. If a file is moved into that folder or an existing file is modified it will create a new note in NixNote with the file name as the title. If NixNote is not running then any new or changed files in this directory will not be imported.

The second type is “Import and delete”. This is the same as “Import and keep” except for two things. The first difference is that the file is deleted when the note is created. The second difference is that if a file exists in that directory when NixNote starts it will import that file and create a note, so that even if NixNote is down a note will be created when it starts.

Both types of imports allow for recursive scanning of subdirectories, but please BE CAREFUL. If you tell it to import/delete it will IMPORT AND DELETE. Sometimes this can be disastrous if you setup a really bad location (like root).

If NixNote is running, it will notice the new (or changed) file as soon as it is written. It might not appear in your note list, however, until the search is refreshed.

Import/export of data

Under the File menu there is an option to export and import notes. Exporting notes creates a NixNote export file with the extension of nnex and contains information about any selected notes. This includes notebooks, tags, & resources.

Restoring a note from a NixNote export does not replace any existing notes and it does not create any notebooks or tags. An imported note is treated as a new note.

Backup/restore of data

The backup & restore process is similar to the import/export process with a few exceptions.

The first difference is that everything is backed up (not just the selected notes). This means that it will take longer to backup a database than to export a few notes.

Restoring from a backup is also different. I only recommend using this when you have an empty database. Restoring will not create any new notes. Any restored notes should have the same GUID, tags, & be in the same notebook as it was when it was backed up. It also will restore notebook, tag, & saved searches.

Linked Notebooks

With a few minor exceptions, NixNote handles shared notebooks in a manor similar to Evernote. You cannot share individual notes or share notes via Twitter or Facebook. To share a notebook, you must instigate the share using an Evernote client.

LaTeX Formulas

NixNote has the ability to add LaTeX formula images. This will only work if you have the “mimetex” program installed. Unlike NixNote 1.x versions, Internet access is not required.

The first way to insert a formula is to type the formula in a note, highlight it, and select “Insert LaTeX Formula” via the context menu.

The second way is to select “Insert LaTeX Formula” via the context menu with no text selected. This will cause a dialog box to appear and you can enter the formula.

Any LaTeX formula is treated by NixNote as a bitmap image. As a result, the image is sent to Evernote where the normal OCR processing is done. Once this is done, you can search for text in the image the same as any other Evernote client. Since other Evernote clients also see this image, you can search on those clients too.

NixNote does allow you to edit a LaTeX formula by clicking on a LaTeX image.

Command Line Options

--accountId=<id_number>

This allows you to have multiple instances of NixNote running at a time. By default, NixNote will always start using the last viewed account and you cannot have multiple copies of NixNote running at time. By passing the –accountId command line parameter, you can force NixNote to start on a specific account and have multiple instances of NixNote runinng at a time.

Frequently Asked Questions

What Evernote features does NixNote support?

NixNote supports most of the same features that Evernote's Windows & Macintosh clients support. The major exceptions are the lack of ability to edit ink notes, only a subset of the search syntax is supported, and it doesn't have the ability to post to Facebook or Twitter.

I start NixNote, but nothing happens. What is wrong with this program? Is it broken?

Most likely you are missing some type of dependency. Open a terminal window and issue the command “nixnote2”. Look for any console messages about missing or incompatible libraries.

Can I do a client-to-client synchronization or synchronize without going through Evernote?

No. NixNote was designed to be a client to Evernote. It currently doesn't have the ability to synchronize directly with other clients. You can, however, do a backup of your database and restore it to another client. It isn't a true synchronization but it can reduce the amount of network activity needed when you start NixNote on a machine for the first time.

Is Evernote giving anything?

Evernote has been extremely supportive in NixNote's development. Any question I've asked has been answered and they've been very helpful with things like encryption and ink notes.

Why don't you support creating ink notes?

Evernote doesn't provide an API for ink notes. In order to support creating ink notes I'd need to reverse engineer their existing file format. That simply isn't something I have the time or ability to do at this point. The only reason read-only ink notes are supported at all is because it pulls the image from the web site. Even Evernote has limited ink note support. Only their Windows clients support creating ink notes. Other clients are read-only just like NixNote.

What about NixNote 1.x?

NixNote 1.x had some limitations that were too difficult to overcome without doing a complete rewrite of the program. Initially I resisted this because starting over was a massive task, however as time wore on it became more obvious that something needed to be done. Some of the features of NixNote 1.x required older SSL libraries that were difficult to find, the libraries were becoming increasingly outdated and causing issues with newer versions of Java, and the simple fact that it was never

initially desigend to be a full client. Eventually the only choice available was to rewrite it.

The old NixNote 1.x ran on OS-X & Windows. Will there be a Windows/Mac version of NixNote 2?

Probably not. I'm not sure how to port the Thrift libraries to Windows and I don't have a Macintosh to compile NixNote. However, it is open source so you can do it yourself if you wish.

Can you implement feature ___?

Maybe. There is a section in the user boards and in the SourceForge project to suggest features. I don't promise anything but many of the current features are the results of people asking. Things like auto-import folders, LaTeX formulas, and viewing ink notes are all things that were added by request.

Can it run on a thumb drive?

Technically there isn't anything preventing it from running on a thumb drive. You'd need to setup your .nixnote directory to point to the thumb drive via a symbolic link and it should work. I would think it would be difficult, however, to make sure the dependencies are installed on any computer you happen to use.

Is there a way to set global hot so I can press Ctrl-F from anywhere & bring up NixNote?

Not at this time. To the best of my knowledge Qt doesn't have any way of getting hotkeys into any window managers. If you know of a way let me know.

How can I help?

There are several ways to help. If you enjoy programming and like looking through awful, poorly documented, difficult to understand code then you can always help improve it by helping to develop it. If coding isn't something you enjoy (or you looked at how it is written and were too frightened to continue) you can contribute by identifying bugs or offering suggestions for improvement.

Can I contribute money?

If you like NixNote enough to pay for it, please take whatever amount you feel it is worth and donate it to your favorite charity. If you think about it, you can send me an email or post on the user forums and let me know.

Development

This section is for anyone who wants to know how NixNote works at a programming level. The goal of this section is not to document every class & method, but to give a general overview of how NixNote works.

Setting up a programming environment

I use Qt Creator and the project files are all on Github. To download everything, do a “git clone https://github.com/baumgarr/nixnote nixnote” and a directory called “nixnote” will be created. Simply open up Qt Creator and open the .pro file.

These are the Ubuntu packages needed for building NixNote. Other distributions will have different package names, but this should give you a good idea of what is needed:

- qtcreator
- qt4-dev-tools
- libpoppler-qt4-dev
- libopencv-dev
- libhunspell-dev
- libboost-dev
- libboost-test-dev
- libboost-program-options-dev
- libevent-dev
- automake
- libtool
- flex
- bison
- pkg-config
- g++
- libssl-dev
- tidy

You will need to have Thrift compiled. If you do not have Thrift compiled, follow these steps

- go to your git repository
- * cd into ./thrift-source/thrift-0.9.1

- Issue "./configure --without-php --without-php_extension --without-python --without-erlang –without-java"

After the config is complete, you should ssee a line similar to this:

"Building C++ Library ..... : yes"

- Issue "make"
- Issue "sudo make install" to install the thrift libraries

Programming Language

NixNote is written in C++ using the Qt libraries.

Database

NixNote uses a sqlite databose for everything except attachments & thumbnails. The database is found in the ~/.nixnote/db-<account> directory. For most people, you will only have one account so it will be ~/.nixnote/db-1. Attachments are stored in ~/.nixnote/db-<account>/dba directory. Thumbnails are stored in the ~/.nixnote/db-<account>/tdba directory.

If you have multiple accounts, they will each have their own ~/.nixnote/db-<account> directory. When an account is deleted the database & al attachments are removed.

Evernote data

Each note, resource, saved search, notebook and tag have a global unique identifier (GUID) assigned to them and they have a sequence count. The GUID is unique among all Evernote accounts, so there should not be any duplicates. This GUID is used to identify these resources across different Evernote clients.

Each Nixnote entry has a LID number. This is a simple sequential number that is assigned when the item is created. The number is NOT unique among different accounts or databases and only exists within a single client. Using it removes the need to update a large number of GUIDs when a note is synchronized.

The synchronization number indicates the current version of the note. When a note is updated on Evernote's servers it gets a higher sequence number. This sequence number is assigned by Evernote (not NixNote). A sequence number is unique within a users account, so no two GUIDs will have the same sequence number. A note which has a sequence number of 1, when synchronized, may next have a sequence number of 243.

This sequence number is used to indicate the position at which you last synchronized. When starting a sync, you pass the highest number you've ever received and it will give you data starting from that point.

Threads

NixNote has several threads of its own that it uses for processing.

The first thread to start is the main (or NixNote) thread. This is the master thread that controls most other threads. There is a restriction that no Widgets (pretty much anything GUI) can exist outside the main thread. This means that any fields that are presented to the user and any updates the user make must pass through this main thread. Doing something foolish in this thread can really hurt performance.

This main thread spawns several other threads:

- SyncRunner thread
- IndexRunner thread
- Three CounterRunner threads

Communication to all threads is handled in two ways.

The first method is by passing methods over a blocking linked list queue. The main thread posts actions into this thread and (when it does) the other client thread will become unblocked and begin doing work. If there is no work for the thread it tries to read an entry from the queue and becomes blocked until a request is made.

The second method of communication is via the Signal API in Qt. Qt permits signals to be emitted and a client in another thread can receive them. This is how the child threads communicate their status back to the main thread in most instances.

Threads should remain active for the life of the program. A timer will wake up every minute and check that the threads are alive. If a thread is not alive it will show a warning to the user and recommend that the program be restarted.

SyncRunner thread and synchronization processing

The SyncRunner thread handles most of the interaction with Evernote. It is the process that deals with synchronizing any notes and resolving any conflicts. Most of the time this thread is asleep and will wake up when the synchronization timer has expired, or when the user clicks the synchronization button to do a manual sync. If the SyncRunner encounteres a problem it will post an error message back to the main thread and disconnect from Evernote.

When the SyncRunner thread is running, a global flag should be set to indicate a sync is in progress. This will prevent the ThumbnailRunner and IndexRunner

threads from running. The logic is that they shouldn't waste CPU doing work for a note that may change in a few moments.

The SyncRunner thread goes through some “interesting” logic to handle everything. This is a rough overview of the flow.

The first thing it does is it tries to validate your userid & password with Evernote. Assuming that is successful, it continues. If it isn't successful it is done and the thread waits for the next request.

The next thing it does is to get the User's account information. This is used to determine if a user is a premium member, their Evernote email address, and their quota usage.

The next information it retrieves is from Evernote is the sync status of your account. This will give you the highest sequence number on your account that Evernote is aware of. If this count is higher than yours then you need to get data from Evernote. If this number is the same as yours then you don't need to download any data.

The sync status also contains a date. This date is a “drop dead” date. If the last time you synchronized is before this date, then Evernote no longer has the individual changes you need to synchronize. If you are in this state, you will need to do a full synchronization.

If you need to get information from Evernote, the first thing you tell it is the highest sequence number you received in the past. This tells Evernote where to begin sending data. Evernote sends data an “chunks”. These chunks can be any size, but I do about 10 records per chunk. There isn't any reason, it just felt like a good round number.

A chunk of data contains any items (notes, resources, tags, notebooks & saved searches) which have changed. It contains records indicating deleted records a well as new & changed records. A chunk also contains the highest sequence number in that chunk. This sequence number is changed so we know where to begin the the next time we get data.

Notebooks, saved searches, and tags are pretty straight forward. Any change from Evernote will be added or updated. If there is a conflict, Evernote wins. For example, if you renamed a notebook on the web interface and you edited it on NixNote, the Evernote name will win and you'll lose the NixNote change.

Note changes are a bit different, since we need to deal with potential conflicts.

If a note hasn't been changed locally, but it was changed remotely then we just update the note in the database, flag it to be reindexed, and move to the next note.

If there is a conflict (a note has been changed locally and on Evernote) then we have a conflict. In this case, the note that was changed in NixNote is given a new GUID by NixNote. It is then moved to a local notebook called “Conflicts”. If a “Conflicts” notebook doesn't exist, it creates one and moves the note into it. It is then the users responsibility to decide what to do with the conflicts. If the user simply moves the note from the conflicts notebook back into a synchronized notebook, it is treated as a new note and will get a new GUID.

The next part of a sync is to look at what has changed locally. When a note, notebook, saved search, or tag is changed NixNote flags that note as “dirty”. Dirty items are items that need to be synchronized.

NixNote goes through its database looking for any records which are flagged as “dirty”. Those records are then sent to Evernote. When the note is sent, Evernote will send back a new sequence number for that record. That sequence number is saved back with the note. If it is a new record, the GUID will change as well, so any references to that GUID need to be updated.

The tricky part about sending changes to Evernote are tags. Before a new child tag can be created, its parent tag needs to exist at Evernote.

For instance, if I were to create TAG1 & TAG2 and set TAG2 as a child to TAG1, neither tag has a valid Evernote GUID so, when I synchronize, I need to be sure to synchronize TAG1 first, get the new GUID, update the parent GUID for TAG2, then finally synchronize TAG2. If there is an error with TAG1 and it can't be synchronized, then TAG2 will not be synchronized either. Most of this is accomplished with lists to keep track of what tags are ready to be synchronized.

Finally, the last thing the synchronization process does is to synchronize linked notebooks. This is essentially the same as synchronizing your own account as described above, except you go against the notebook owner's shard. The other major difference is that changes to tags & notebooks in linked notebooks are not synchronized (you can't update another person's notebooks or tags). This is an Evernote restriction. NixNote will let you change them, but that change is only on that system and it isn't synchronized. Tags & notebooks are stored a little differently in the database to denote that they are linked instead of tags you created.

Each linked notebook is a different sync. If you have 10 linked notebooks, the sync process is run against each notebook individually. The initial sync of a linked notebook seems to be very slow. It looks like it sends the data in chunks (the same as synchronizing your account the first time), but it will send empty chunks if you are not authorized for any data in that particular chunk. For example, if I can receive a chunk that contains nothing.

IndexRunner Thread

NixNote maintains a separate database for searching. Adding words to this database is handled by the IndexRunner thread.

The first thing the IndexRunner will do is to determine if any notes need to be indexed and it will parse them and add individual words to the database. Note resources are not done at the same time as the contents of a note.

Once all notes are done it will index note resources (attachments and OCR data). The OCR data is received from Evernote and is in XML format. Once the OCR data is added it will then handle indexing their attachments.

CounterRunner Threads

The notebook, tag, & trash lists in the GUI maintain a count of notes. The actual counting, however, is done in a separate thread to improve user performance. One thread counts notebooks, one counts tags, & the other counts notes in the trash. All three of these threads use the same threads and are pretty similar in their function.

Once a count is complete, the total is passed back to the main thread via signals. If a thread is counting and another count request comes in, the existing count is canceled and the counting is restarted. For example, if it is counting and you switch notebooks a new count is ordered and the existing count is aborted. This is done to prevent meaningless counts and improve user responses.

Program Directory Hierarchy

The NixNote GIT repository contains the following hierarchy:

certs → Digital certificates used to authenticate Evernote.

communication → Classes used for communication with Evernote. Typically (but not always) called from the syncrunner thread.

dialog → popup dialog boxes.

evernote → Clasess provided by Evernote that were generated by Thrift.

filters → Classes used to filter database records from GUI classes.

gui → Most of the classes that provide visual elements to the users.

html → Classes used to format notes from HTML to Evernote's ENML markup data and vice versa.

images → Icons, PNGs & other picture elements.

java → Java classes used to encrypt & decrypt text (RC2 only).

libencrypt → Obsolete.

logger → Handles messages to the application log file & the user's log file.

man → Unix man files.

models → Classes used for Qt's Model/View structure.

oauth → Classes used to popup the initial Evernote window granting an authorization token.

package_scripts → Scripts used to create the rpm, deb, & tar files.

qss → Style sheets for Qt's theming support. Not currently used.

reminders → Classes used to support reminders within notes.

settings → Classes used to maintain program & user settings. IT also helps maintain where the different directories used by Evernote are.

spell → Contain's a user's custom dictionary for hunspell.

sql → Most of the classes that are used to maintain database records. There are a few other locations that access the database directly, but most of the DB I/O happens here.

threads → Subthreads syncrunner, counterrunner, & indexrunner.

thrift → Thrift generated C++ files.

thrift-source → Source for Apache Thrift. Depending on if you have Thrift installed, you may not need this.

translations → Qt Linguist language translations.

utilities → miscellaneous classes used for various utilities.

watcher → Classes that deal with auto-import of notes.

webcam → Classes that deal with webcam notes and interfaces.

xml → Classes used for nnex & enex imports & exports.