Spark Reader manual

Table of Contents

[Notice 1](#_Toc471554538)

[Starting/closing the program 2](#_Toc471554539)

[Setting up a text hooker 2](#_Toc471554540)

[The main interface 3](#_Toc471554541)

[Navigating the interface 3](#_Toc471554542)

[Word right click menu 4](#_Toc471554543)

[Minimizing to tray 4](#_Toc471554544)

[Word splitting 4](#_Toc471554545)

[Selecting definitions 5](#_Toc471554546)

[Known words list 5](#_Toc471554547)

[Importing known words from Anki 5](#_Toc471554548)

[Importing words from other sources 6](#_Toc471554549)

[Exporting words 6](#_Toc471554550)

[Heisig Kanji integration 6](#_Toc471554551)

[The custom dictionary 7](#_Toc471554552)

[Settings 7](#_Toc471554553)

[Troubleshooting 7](#_Toc471554554)

[Multiplayer (WIP) 7](#_Toc471554555)

# Notice

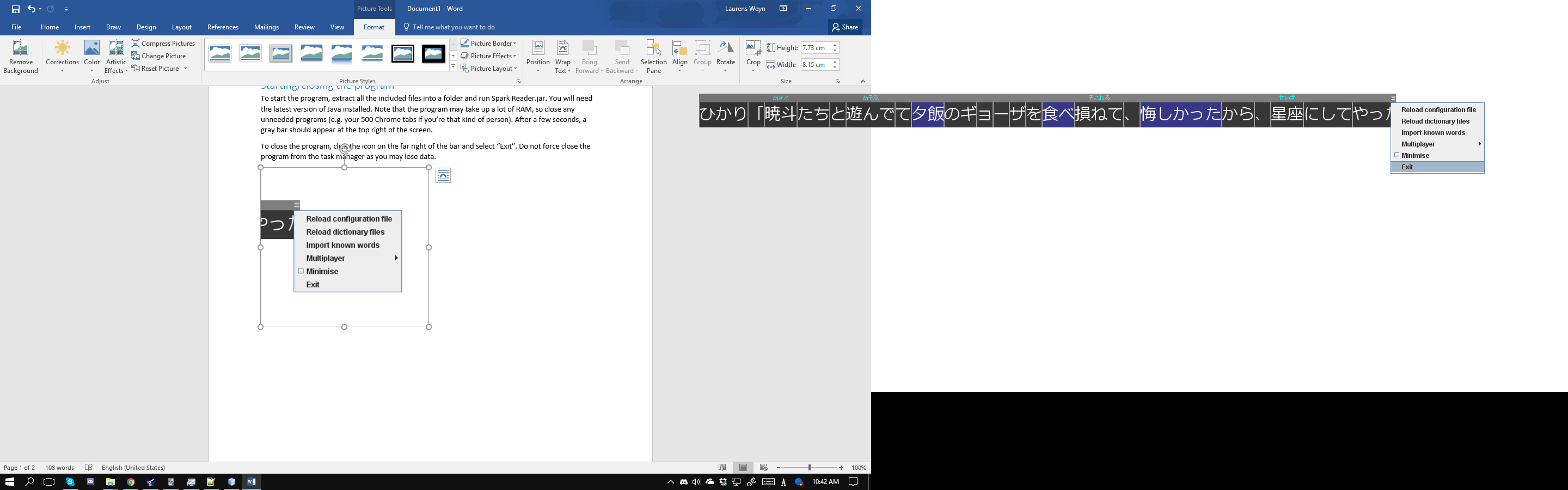
This software is unfinished. Things are not guaranteed to work or be correct, and there’s probably many alternatives far better than this program available. I appreciate you for trying it out though!

This program comes with the EDICT file from The Electronic Dictionary Research and Development Group. See more info [here](http://www.edrdg.org/jmdict/edict_doc.html).

Want to help with (Java) development? The project is on [GitHub](https://github.com/thatdude624/Spark-Reader).

Made by Laurens Weyn

# Starting/closing the program

To start the program, extract all the included files into a folder and run Spark Reader.jar. You will need the latest version of Java installed. Note that the program may take up a lot of RAM, so close any unneeded programs (e.g. your 500 Chrome tabs if you’re that kind of person). After a few seconds, a gray bar should appear at the top left of your screen.

To close the program, click the icon on the far right of the bar and select “Exit”. Do not force close the program from the task manager as you may lose data.

If the window is too big or small (I recommend making it the same size as the game window), or you don’t like the colors or fonts, please see the “Settings” section

# Setting up a text hooker

Spark Reader does not come with a text hooker (yet, later versions may add one). You will need to use your own. If you have ITHVNR or whatever, you probably already know how to set this up.

If you are using VN Reader, open the Option popup menu to the left of the game and turn off everything except “Text to clipboard”. Turn on transparent mode as well on the left menu.

Spark Reader isn’t limited to VNs of course. Any Japanese text you copy to the clipboard will automatically be loaded.

# The main interface

When text is loaded, it will appear on screen. Clicking on a word will open a popup showing the definitions of that word. (note that the selected word is incorrectly split, we’ll get to that later)



1. A found word. The gray lines to the left and right indicate where the splitter determined the word probably starts and ends (see the “word splitting” section for more details)
2. The reading. For verbs this is left in dictionary form. The reading is taken from the currently selected definition (see “selecting definitions”)
3. Known words are marked in blue, and their furigana is hidden (see “Known words list”)
4. When a word is selected, the furigana over it is replaced with the currently selected definition, and how many definitions are available.
5. If the word was deconjugated or altered in some way, this line appears showing the “dictionary form” of the word and how it was achieved.
6. This shows the EDICT tags for this definition. n means noun, v1 means ichidan verb and v5\* means godan verb. See the full list of tags and their meanings on the “lexicographical details” section of [the edict documentation](http://www.edrdg.org/jmdict/edict_doc.html).
7. The definition text. If this word has multiple readings/spellings, they will be listed above this definition text as well.

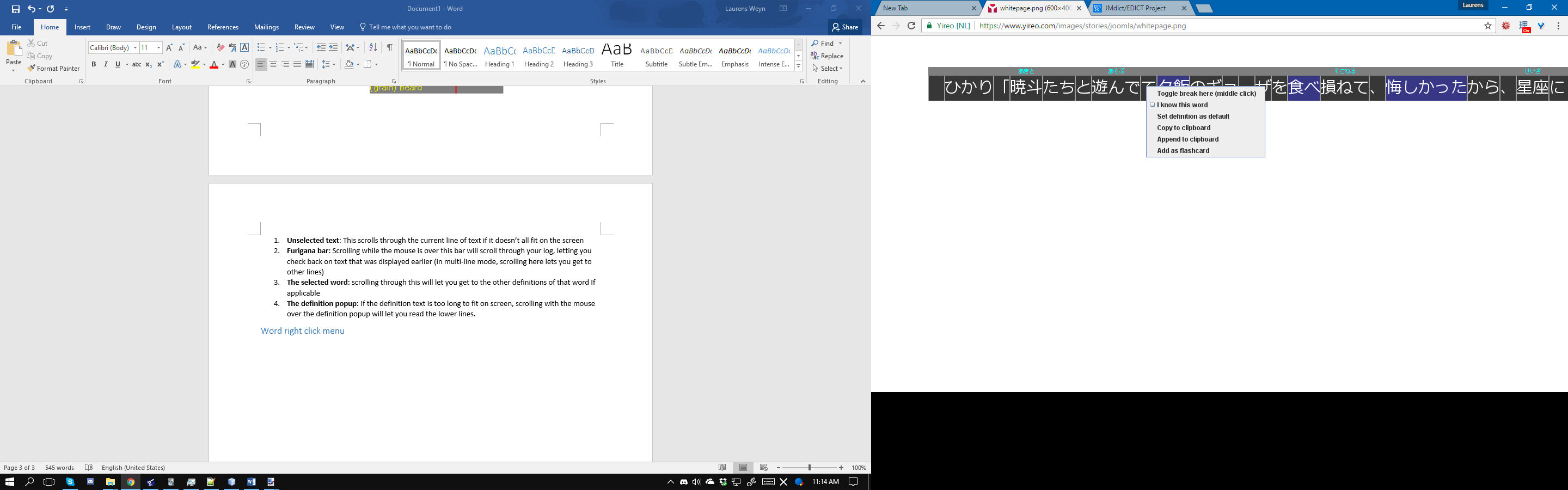
## Navigating the interface

A lot of UI interactions use the scroll wheel. If your mouse doesn’t have a scroll wheel, you’re gonna have a mad time. Here are the different places you can scroll:



1. **Unselected text**: This scrolls through the current line of text if it doesn’t all fit on the screen.
2. **Furigana bar**: Scrolling while the mouse is over this bar will scroll through your log, letting you check back on text that was displayed earlier (in multi-line mode, scrolling here lets you get to other lines).
3. **The selected word**: scrolling through this will let you get to the other definitions of that word If applicable.
4. **The definition popup:** If the definition text is too long to fit on screen, scrolling with the mouse over the definition popup will let you read the lower lines.

## Word right click menu

Right clicking words will bring up a right click menu. Most of these options are explained in their respective sections.

“Copy to clipboard” will replace the clipboard contents with just that word (without replacing the displayed text), for if you want to look up a word in another dictionary. If you copied a word and right click the word to the right of it, a new option will appear: “Append to clipboard”. This adds that word to the clipboard along with the previous one, for when you want to look up a word that isn’t being split correctly.

## Minimizing to tray

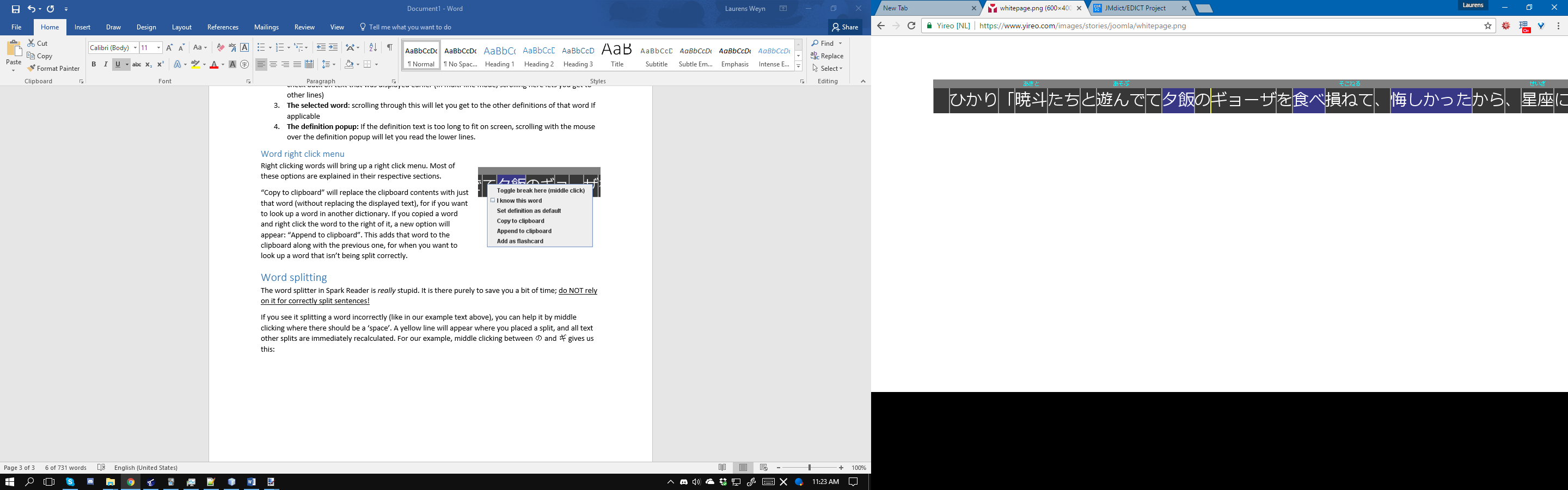
If the UI is temporarily in the way, you can minimize it to the system tray through the context menu on the top-right.

Doing so will hide the interface. It will automatically reappear when a new line of Japanese text is detected (this can be turned off in the settings file), or when the tray icon is double clicked.

# Word splitting

The word splitter in Spark Reader is *really* stupid. It is there purely to save you a bit of time; do NOT rely on it for correctly split sentences!

If you see it splitting a word incorrectly (like in our example text above), you can help it by middle clicking where there should be a ‘space’. A yellow line will appear where you placed a split, and all text other splits are immediately recalculated. For our example, middle clicking between の and ギ gives us this:



Alternatively, you can right click and select “toggle break here” to do the same thing. Any new known words will also be highlighted and furigana will be updated as well.

# Selecting definitions

Definitions are sorted by various criteria internally to try and put the best definition at the front. Unfortunately, this will not always work correctly, resulting in incorrect furigana.

To change the furigana, click on the word and scroll to the right definition (If the definition doesn’t exist [e.g. it’s a character’s name] see “custom dictionaries”) When the definition is closed, the furigana will be updated.

If a common word is getting assigned a rare definition, scroll to the right one and then right click the word and select “Set definition as default”. This will put that definition at the front the next time you encounter the word.

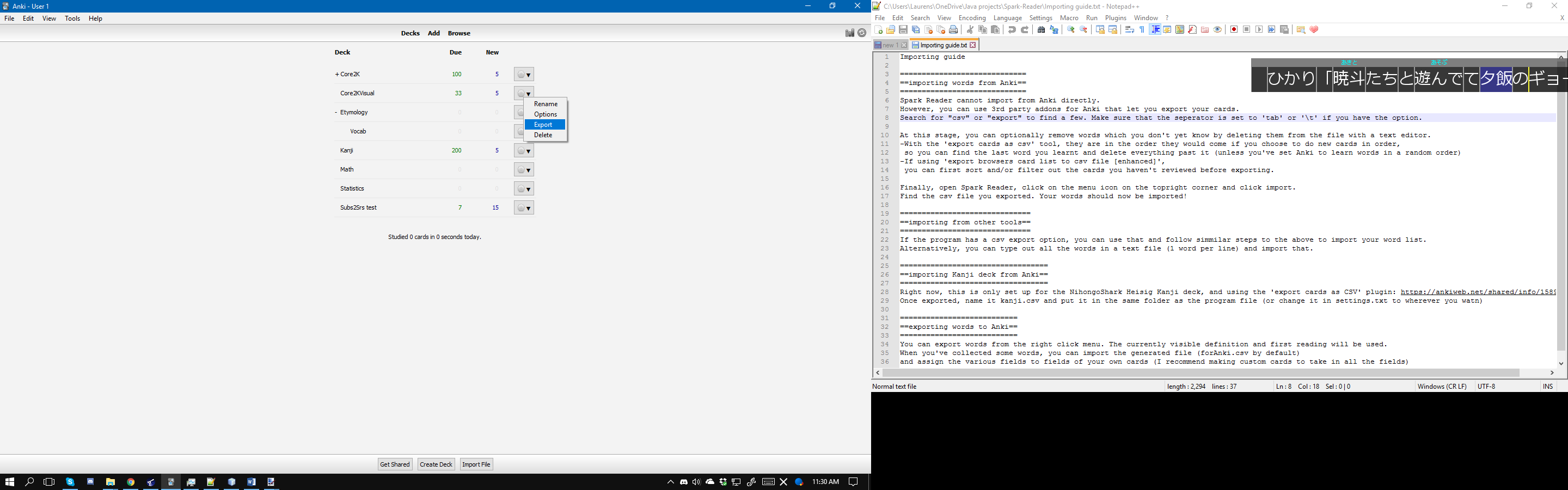
# Known words list

It can be detrimental to learning to rely on furigana all the time and to keep looking up words you should already know. When you encounter a word that you should know, right click it and select “I know this word” to hide the furigana and highlight it in blue (so you feel slightly more guilty clicking on it to read the definition).

It’s a pain to do this for every word you already know of course, so if you have some sort of database of words you know, you can import them into Spark Reader. I’ll demonstrate with the most common one: Anki

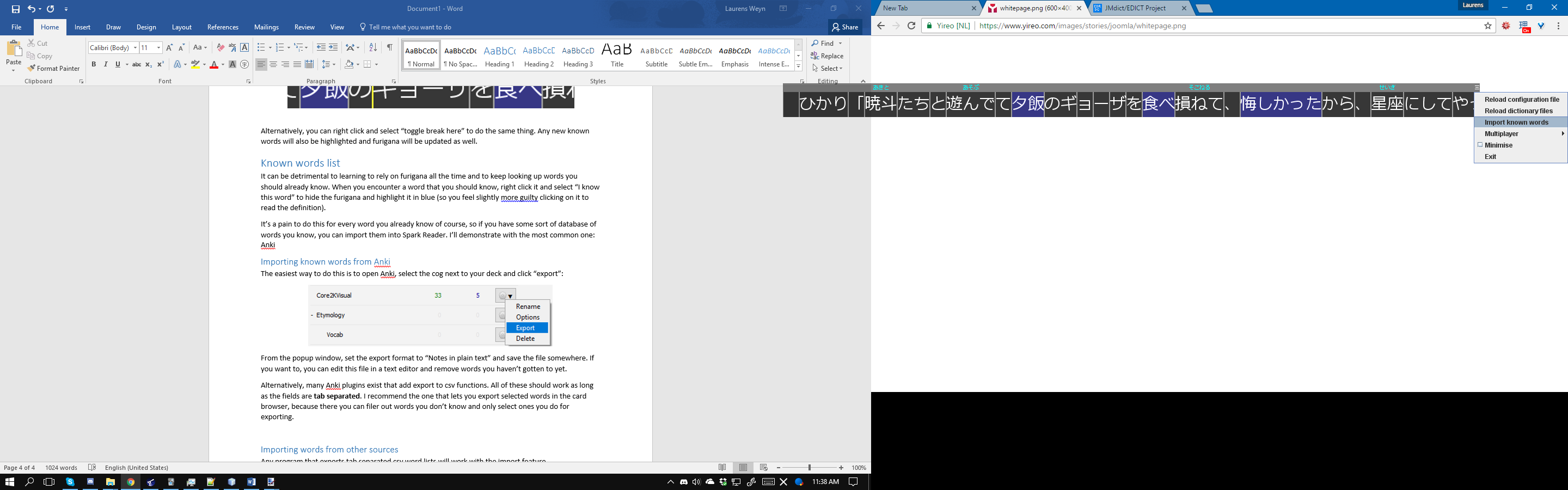
## Importing known words from Anki

The easiest way to do this is to open Anki, select the cog next to your deck and click “export”:



From the popup window, set the export format to “Notes in plain text” and save the file somewhere. If you want to, you can edit this file in a text editor and remove words you haven’t gotten to yet.

Alternatively, many Anki plugins exist that add export to csv functions. All of these should work as long as the fields are **tab separated**. I recommend the one that lets you export selected words in the card browser, because there you can filer out words you don’t know and only select ones you do for exporting.

In Spark Reader, open the menu and select “Import known words”, and find the file you just exported from Anki in the popup window.

## Importing words from other sources

Any program that exports tab separated csv word lists, or a plain text word list with a new word on each line, will work with the import feature. You can also find a list of words (or an Anki deck) with the 2000 most common Japanese words or something on the internet and import that.

# Exporting words

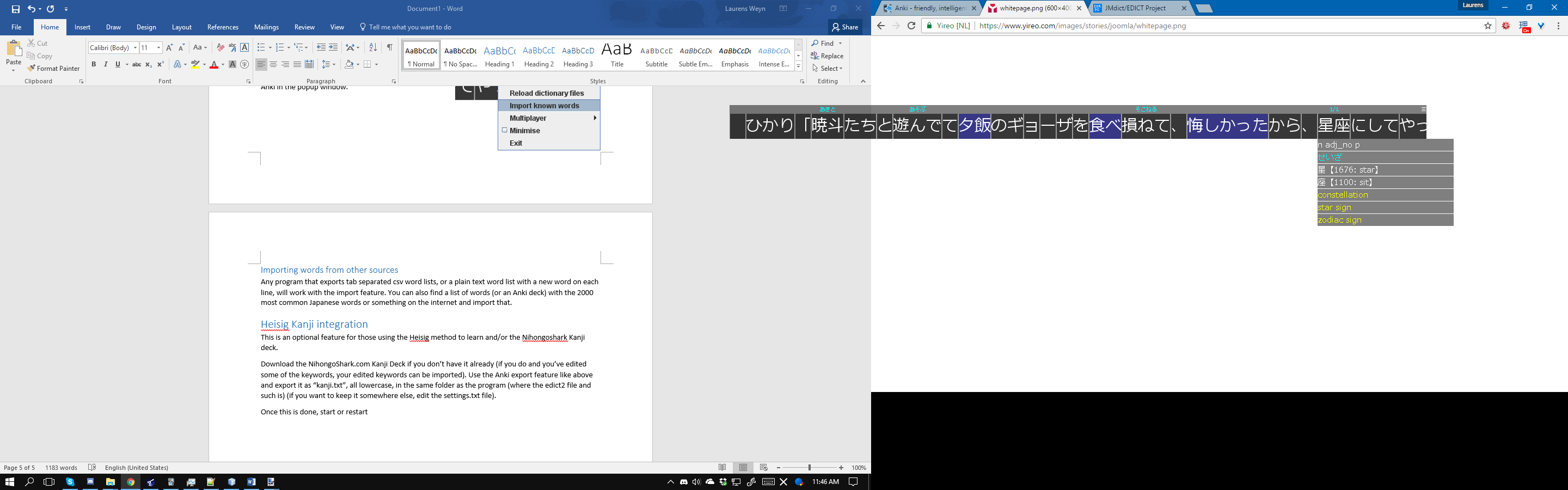
If you find a new and interesting word, you can export it for adding to an Anki deck later. First scroll to the right definition, then right click the word and select “Add as flashcard”.

All words are added to a file called “forAnki.csv” by default. When you have collected enough words, open Anki and go to File-Import (or press CTRL-I) and navigate to the file. Ensure Anki correctly detected that it is **tab separated**, if not, change it or it won’t work properly.

You may want to make custom flash cards for imported words so all the fields can be mapped, which is beyond the scope of this manual. When asked about field mapping, this is what the fields of exported cards are:

1. The word
2. The reading if applicable (furigana)
3. The definition selected when the word was exported
4. EDICT tags, space separated
5. context, or the full line from the VN the word was taken from
6. Kanji details (see “Heisig Kanji integration” to enable this feature, otherwise it will always be blank)
7. User comment that you can optionally add in when exporting the word. You can use this to hint at who was speaking to who, or provide more context than just the line of text that you would otherwise forget when you import them later.

# Heisig Kanji integration

This is an optional feature for those using the Heisig method to learn and/or the NihongoShark Kanji deck.

Download the NihongoShark.com Kanji Deck if you don’t have it already (if you do and you’ve edited some of the keywords, your edited keywords can be imported). Use the Anki export feature like above and export it as “kanji.txt”, all lowercase, in the same folder as the program (where the edict2 file and such is) (if you want to keep it somewhere else, edit the settings.txt file).

Once this is done, start or restart Spark Reader and the Kanji keywords and Heisig numbers will now show up in the definition window.

# The custom dictionary

You will inevitably encounter names in text that won’t be in the dictionary, or made up words. For words like this, you can add them to a custom dictionary to stop the automatic splitter from getting confused and so it can show correct furigana.

Right now, there is no GUI for adding custom words… so you’re going to need to put them in the customDict.txt text file yourself. The format is the same as the EDICT format:

Spelling1;Spelling2;Spelling3 [reading1;reading2;reading3] /(tag1)(tag2)(tag3)def1/def2/def3/code/

You can have any number of spellings or readings, and readings are optional for kana only words. Most of the time, the only tag you’ll need is (n) for nouns. Any number of definitions lines can be added, and the last line of the “definitions” is used as an internal ID code, set this to something random.

Please try follow the format exactly or things might not work right (When the word adding GUI is made you won’t have to worry about this). For this reason, 2 custom words have been added to the dictionary already so you can see how to format your own.

# Settings

If you don’t like the fonts, colors etc, you can change almost all of it in the settings.txt file. The file itself contains information on how each option works, and their names should make it somewhat easy to see what they do.

When you’ve changed a setting, press CTRL-S to save and select “Reload configuration file” within Spark Reader to see your changes, or restart the program.

It’s recommended to change the width of the program to the width of the game you are reading through.

# Troubleshooting

Clicks are going ‘through’ Spark Reader to the program/game below it

If you are having this problem, go to the settings.txt file and change *takeFocus* to true instead of false.

Issue not listed here/wasn’t solved

Please report it on the [GitHub issue page](https://github.com/thatdude624/Spark-Reader/issues).

# Multiplayer (WIP)

*Note: this feature is completely broken in the current beta, don’t even bother trying it. This just describes what it’s* ***supposed*** *to do.*

If you have friends who you want to read with or you want to organize some sort of group reading session, Spark Reader’s multiplayer mode can help everyone stay on the same page, quite literally.

One player has to host the game, by opening the context menu and going to Multiplayer->host. This player has to announce his/her IP address to the other players who can then join. If the other players are not on the same LAN, the host has to port forward port 11037 for multiplayer to work. The host is used as the “reference point” for all of the clients.

It’s recommended that everyone connects and starts on the same line of text at the beginning of the reading session. Whenever a player advances a line, they are told how far ahead or behind they are from the host in number of lines. The host meanwhile can see how many clients are ahead or behind their current line.

This is all multiplayer does right now (*it doesn’t even do that because it’s still broken*). If players are not in the same room, a voice/text chat system like Discord is recommended to be used alongside Spark Reader.