

Table of contents

Table of contents	1
Yarn Spinner basics	2
Before you start	2
THINGS YOU SHOULD NEVER DO	3
Crash course	3
Things to remember	9
Oh no! I saved the file as json *accidentally*	12
Unity version control	12
Updating your branch to be in sync with the Yarn dialogs	12
Preparing for worst case	13
Convert to Twine	13
Lose contact / slow contact: Allan	13
The nitty-gritty	14
Git commands for putting the dialogs in Unity	14
Renaming the git repository	14

Yarn Spinner basics

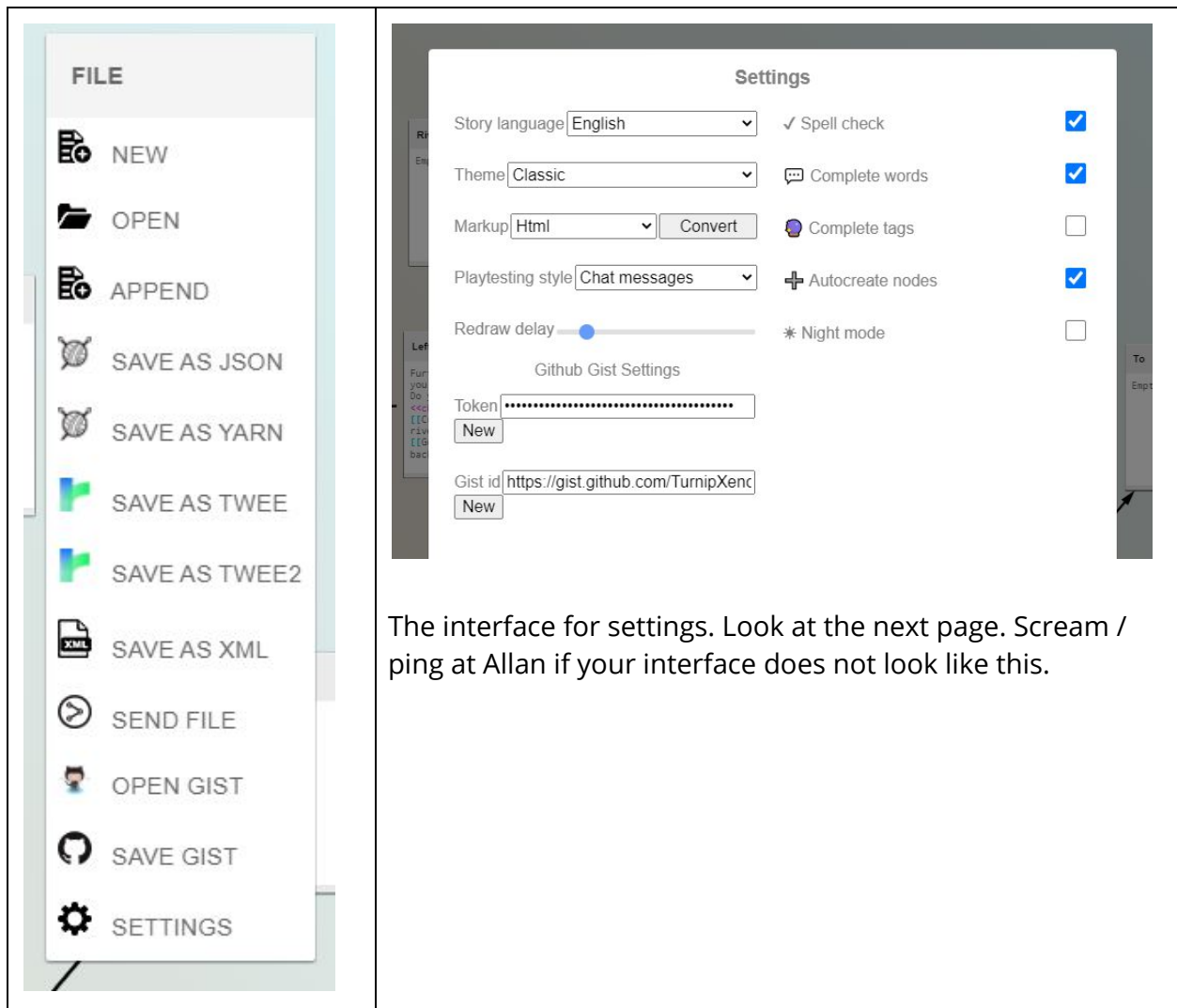
Link to Yarn Editor browser editor: [Yarn](#)

- You can use this on your phone (or table). But, do you really wanna do that? LOL
- The browser editor is sufficient. If you find it really slow, go ahead and download the .exe version for Windows or .dmg in macOS.

Link to Yarn Editor navigation overview: [Using the Yarn Editor](#)


Link to Yarn Editor github (for the nerds): [YarnSpinnerTool/YarnEditor: A tool for writing interactive dialogue in games!](#)

Before you start



The interface for settings. Look at the next page. Scream / ping at Allan if your interface does not look like this.

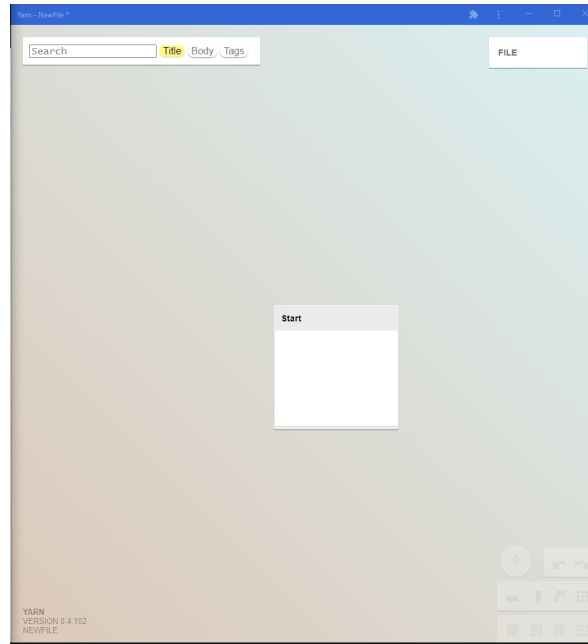
This is the interface so we could get the same frame of reference when I'm giving instructions. I'll highlight (not bold) things you NEED to do in the **SETTINGS**.

- Turn off  **Complete tags**, that one is very buggy when you're trying to create branching dialogues.
- Changing the theme will change the layout, so before you change that, do the setting up first.
- Change **Markup** to **Html**.
- I highly recommend using **Chat messages** for **Playtesting style**. You don't see the previous texts in **Npc bubbles**.
- Turn the redraw delay really low (towards the left). If something's not working with your device, turn up redraw delay.
- Copy paste this in your **Github Gist Settings: Token**:
 - [redacted]
 - Please do not share the token. That's connected to my account. LOL We cannot attach the save location for Yarn into something we can all share, sadly.
- Copy paste this in your **Github Gist Settings: Gist id**:
 - [redacted]

THINGS YOU SHOULD NEVER DO

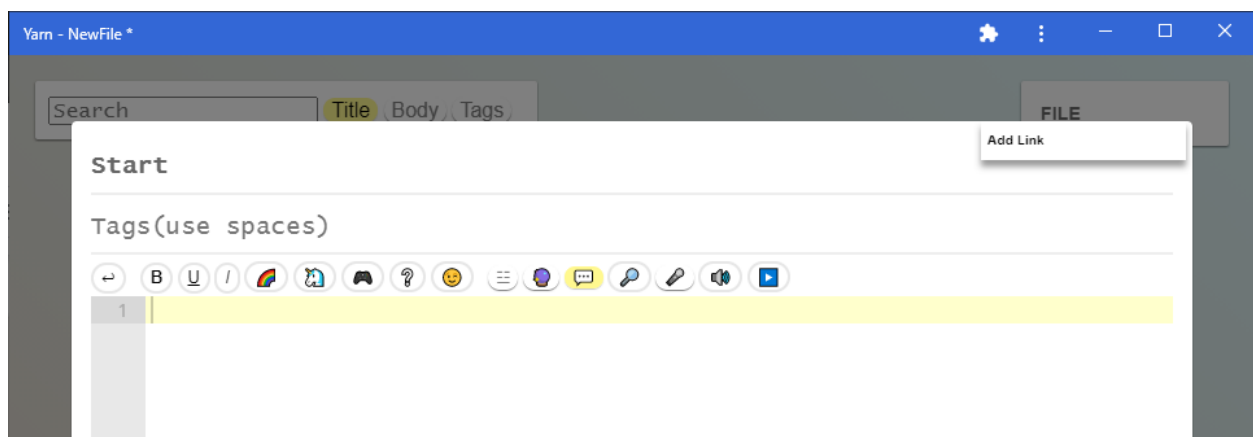
- Don't put # (hashtags / pounds / sharps / number signs). If you see it anywhere, delete it. Please LOL

Crash course



Sadly, there are little tutorials about the editor itself. All the tutorials are for setting things up for programmers. Here's an explanation about everything you need to know in YarnSpinner.

Click on the notepad looking thing. This is called a node. In Twine, this is called a passage. A node can contain several texts, and they can branch off into other nodes. If you double click this, you will get this interface.



You can hover your mouse on the icons and find out what they do. Anyway, if the 🌌 crystal ball icon is highlighted, toggle it off. It's still buggy.

In order, the icons are:

- **B** Bold highlighted text
- U Underline highlighted text

- / Italicize highlighted text
- 🌈 Change font color of highlighted text
 - IF YOU WANT TO USE THIS: REMOVE THE # SIGN
- 🦋 Add inline image / sprite (does not work in our case unless we convert this to Twine)
 - If you want to add sprites / images inline, this is the format:
`<sprite="assetName" name="spriteName">`
- 🎮 Command: you can use this to make branching dialogues based on certain conditions.
 - Check out for a brief overview [Expressions and Variables](#).
 - Everything you need to know about using YarnSpinner can be found here: [Syntax Quick Reference](#).
 - You can branching dialogues based on conditions using


```
<<if $gregg_day_intro is 0>>
  Gregg: Sup duder.
  Mae: hey.
<<endif>>
```
 - You can set up facts or variables. This is useful if you have choices that have long term consequences.


```
Mae: Oh god. Steve Scriggins.
<<set $suspect_steve to 1>>
Lori: Yeah. Him.
```
 - Programmers can also make custom commands so if you have anything you want to happen by doing commands, like `<<shakeScreen>>` or `<<playSound funnySound>>` (the latter one has to be implemented)
 - Check out what custom commands are planned, and which ones are already implemented at [Programming documentation](#)
- ? Choice / Link is when the player is given a choice. It looks like this **[[The option the player sees|TheNameOfTheNodeOrPassagWeGoTo]]**
- 😊 Emoji
- Line counter
- 🌐 Turn this off please LOL
- 💬 Auto-completion
- 🔍 Find text
- 🎤 Transcribe: As long as it's highlighted, you can speak and it's all going to be transcribed. That depends on how good your device is at transcribing your voice.
- 🔊 Hear text

- ▶ Play / preview / debug from the current node

During play mode:



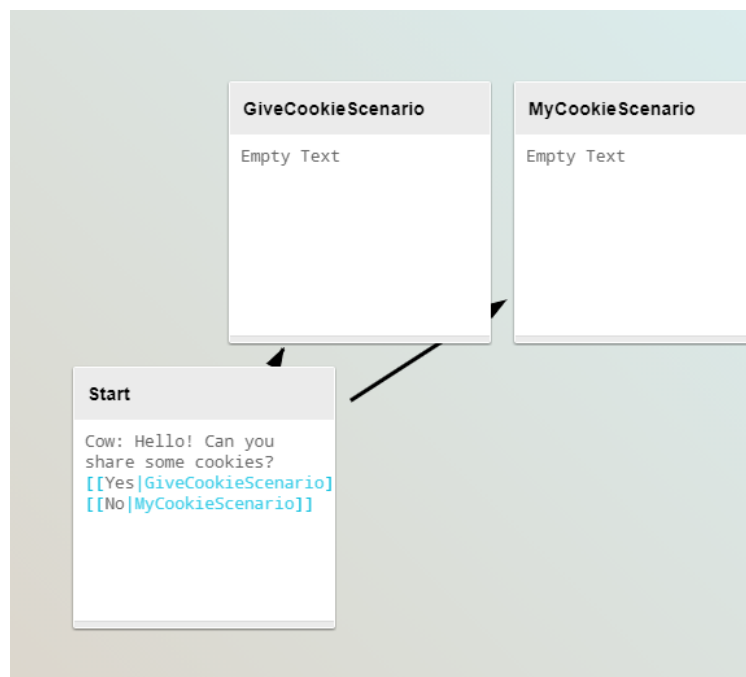
Let's make a story about a cookie.

Try pasting this on your first node.

```
Cow: Hello! Can you share some cookies?  
[[Yes | GiveCookieScenario]]  
[[No | MyCookieScenario]]
```

Try going out of the node by clicking anywhere outside the node. You should now see the two nodes that were automatically created for you.

Use the middle scroll thingy on the mouse to move around the screen.



Let's go to MyCookieScenario node. Put this text in it:

```
Cow: Can I buy the cookie from you? 🍪  
[[Yes | CookieBought]]  
[[No | CookieNotBought]]
```

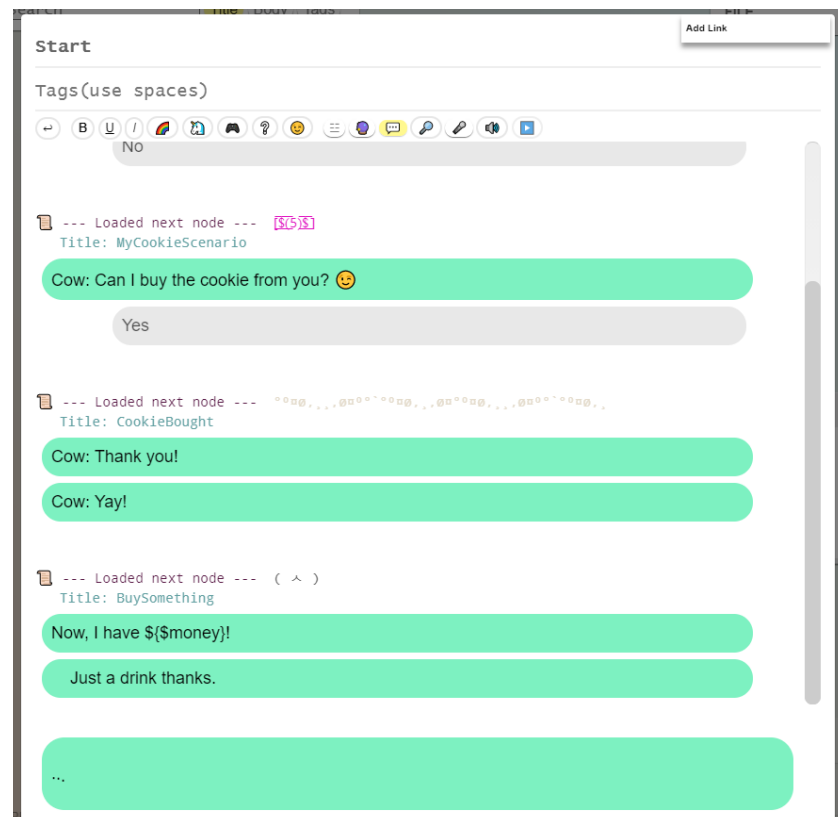
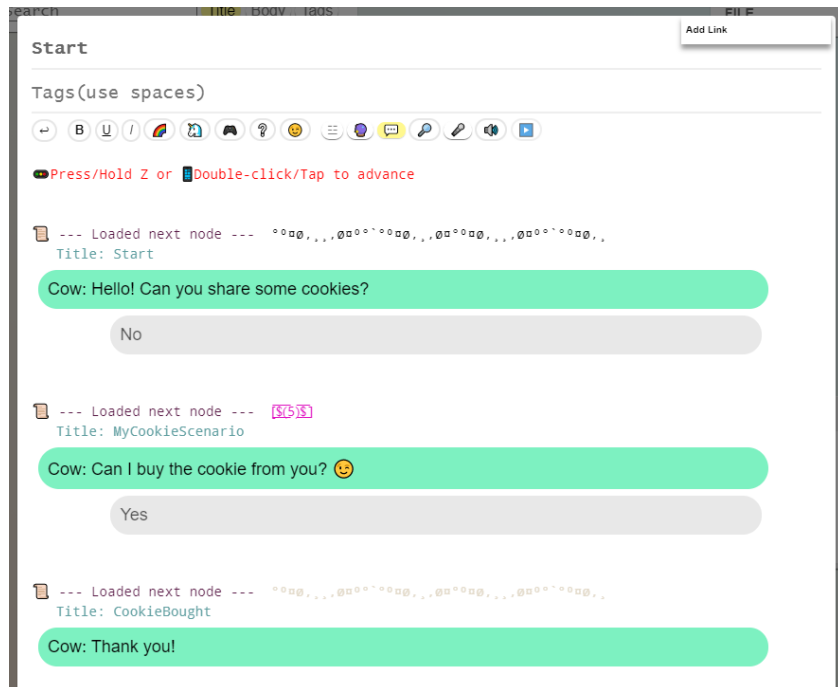
Go to CookieBought node, and paste this:

```
<<set $money to 1>>  
Cow: Thank you!  
<<set $cow_friendship = 1>>  
Cow: Yay!  
[[Continue | BuySomething]]
```

You can see that now, your money is equal to 1. cow_friendship is equal to 1. You can use the variables, as we call it, to make certain branching dialogues. Don't forget the dollar sign before every variable. Let's go to BuySomething.

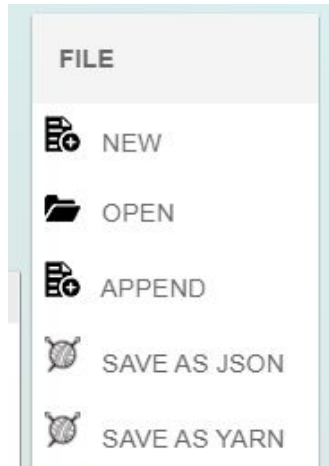
```
Now, I have ${$money}!  
<<if $money > 1>>  
    I would like a horse please.  
    <<set $money to $money - 2>>  
    <<set $hasHorse to true>>  
<<elseif $money eq 1 >>  
    Just a drink thanks.  
    <<set $money to $money - 1>>  
<<else>>  
    Drat, I can't afford anything.  
<<endif>>
```

When you try running or playing YarnSpinner starting from the Node called Start. You should see this.



As you can see, “Now, I have \${money}!” should appear as “Now I have \$1!” In game it does, but in YarnSpinner it does not.

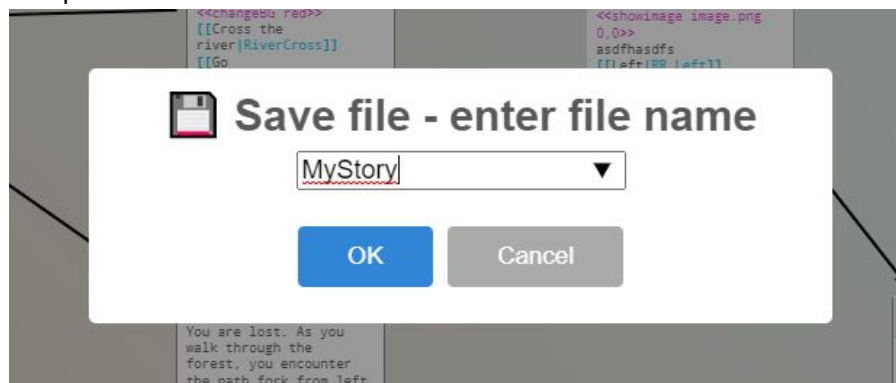
Now here's the weird part in YarnSpinner. Before you save your file for the first time, go **Save as Yarn**. Otherwise, it will be saved as json, which we do not want.



Then, **Save Gist**.



You will be prompted to save.



Name your file, and press OK. That's it. Now you have your story. You can always save again by overwriting your own story.

You can open other stories by going to Open Gist, and selecting the appropriate story in the dropdown menu, then clicking okay.

Things to remember

- Do you want to go ham in text formatting. Check this out: [Rich Text, TextMesh Pro Documentation](#). You can do crazy stuff here. Although, it won't appear on YarnSpinner. Here's a sampling of what they have:

19 Font Size

You can adjust the font size of your text at any time. You can specify the new size in either pixels, font units, or as a percentage. Pixel adjustments can be either absolute or relative, like +1 and -1. All relative sizes are based on the original font size, so they're not cumulative.

```
<size=100%>Echo <size=80%>Echo <size=60%>Echo <size=40%>Echo  
<size=20%>Echo
```

Echo Echo Echo Echo Echo

Adjusting size.

22 Strikethrough and Underline

You can add an additional line that runs along your text. Underline draws the line slightly below the baseline. The vertical offset is defined by the [font asset](#). Strikethrough places it slightly above the baseline.

The <s>quick brown</s> fox jumps over <u>the lazy dog</u>.

The ~~quick brown~~ fox
jumps over the lazy dog.

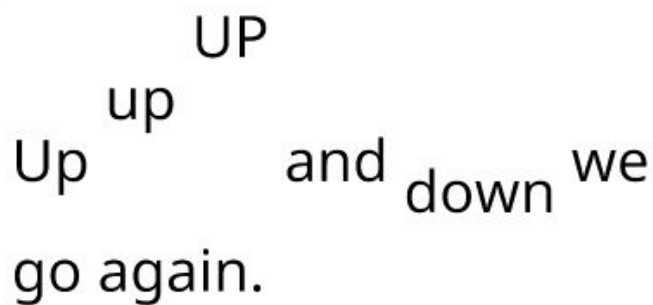
Strikethrough and underline.

25 Vertical Offset

`voffset` gives the baseline a vertical offset. You can use pixels or font units and it's always relative to the original baseline. The closing tag resets back to the original baseline.

The line height is adjusted to accommodate the displaced text. If you don't want that, you can manually adjust the [line height](#).

```
Up <vooffset=1em>up <vooffset=2em>UP</vooffset> and  
<vooffset=-0.5em>down</vooffset> we go again.
```



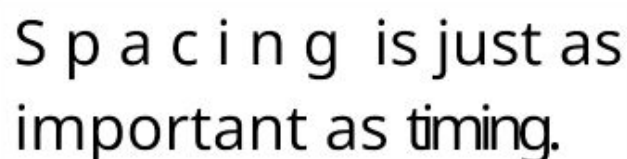
Vertical offset.

5 Character Spacing

`ospace` allows you to adjust the character spacing, either absolute or relative to the original font. You can use pixels or font units. Postive adjustments push the characters apart, while negative adjustments pull them together.

The closing tag reverts back to the font's normal spacing.

```
<ospace=1em>Spacing</ospace> is just as important as <ospace=-0.5em>timing.
```



Character spacing.

- When creating a new file, always **Save as Yarn** before doing **Save Gist**.

- When you want to open a file, always do **Open Gist**.
- Remember to always save. We can always recover files you have overwritten, but we can never recover files you did not save.
- Remember: always open the yarn file and NOT the json file if it exists.
- The dialog currently can only show one line. I can add a feature if you want to show more than one line at a time.
- You can call story passages or nodes from other files. So, be careful about naming your story passages or nodes! Make your story passages / nodes / titles as descriptive as possible. Do not use non-alphanumeric characters except for period or dot.
- We have custom commands so you can use those, too. In YarnSpinner, they don't show up. In the game they do show up! Currently, we have functions like <<doPuzzle puzzleName>> to make a puzzle called puzzleName appear, <<playAudio audioName>> to play an audio called audioName to play, and many more at

Oh no! I saved the file as json *accidentally*

Starting from that point, your file will be saved as json. We do not want that. To force saving again as a yarn file, exit application, reopen the application, and save as yarn. Then, save your file. Ping Allan if something's lost because of this issue.

Unity version control

None of the Git GUI options support git subtree. Using git subtree allows anyone to easily get the repo, and run the files. With submodules, the burden is on the user. With git subtree, the burden is on the maintainer.

I think this part is complicated. I will always update the master branch and dialog_refresh branch to be up to date with our dialogues. Otherwise, I'm putting how to update the branch dialog_refresh below, if you want to try it out or do the updating yourself.

Updating your branch to be in sync with the Yarn dialogs

If you think you're having trouble, you can always just merge with master since I will always maintain master to be updated with the dialogs.

Here's the commands we use to update our dialogs.

```
git remote add yarndialog
git checkout dialog_refresh
https://gist.github.com/634bd8024ef23ecea396539316ea7efd.git
git fetch yarndialog master
git subtree pull --prefix=Assets/Dialogs/ yarndialog master --squash
```

Explanation:

Line 1: Do this during your first time uploading the repository. What this does is create a shorthand version to make updating the dialogs easier. You only need to do this ONCE.

Line 2: Checkout a different branch where we pull the changes in dialog

Line 3: Fetch, well, fetches all the new changes from the Yarn dialogs the writers are working on. They don't really have any other choice but to work on master in gist, so that's what the master is for.

Line 4: Git pulls the latest changes from the dialogs.

Preparing for worst case

Convert to Twine

We can always convert back to Twine if things don't work out. We can do this by saving our file as Twee2, and using Twee2 compiler to convert it back to Twine. This will only happen in the worst case scenario. For that, you will need to follow the instructions in installing from <https://dan-q.github.io/twee2/install.html>. Then, you can do this in your terminal:

```
twee2 build your-file.tw2 the-name-of-your-output-file.html.
```

Since YarnSpinner is very simple, I don't really expect it to have a lot of issues. But, issues will not be avoidable. We can always clean it up by importing it back to Twine.

Lose contact / slow contact: Allan

I was thinking about what could be the consequences of using gist and that is that the person who only has the control to a gist file is one person. No form of collaboration exists. In this case, I'm the only one who has deleting and merging privileges. The only thing everyone else can do is save and create files. When we have made sufficient progress in Gist already, I guess I'll post my GitHub credentials here. Anyway, this can be used when there is an emergency and you really need to edit the gist files. Maybe in the case of requiring conflict merges or lost files. You will need these credentials to use GitKraken /

Github Desktop to edit files in the dialogs, too. You may also need it for merging things and getting information from the dialog Gist.

- Username: **[redacted]**
- Password: **[redacted]**

The nitty-gritty

[2020/10/16] I think it's very easy to scale bigger, and art can get ugly and lose info when it goes smaller. So I will assume that the screen is 2224 x 1668. In Unity, I have arbitrarily assigned that one unit as 100 pixels (default image conversion in Unity). The visible screen is 22.24 x 16.68 Unity units / meters.

Git commands for putting the dialogs in Unity

```
git remote add yarndialog  
https://gist.github.com/634bd8024ef23ecea396539316ea7efd.git  
git fetch yarndialog master  
git subtree add --prefix=Assets/Dialogs/ yarndialog master --squash
```

Explanation:

Line 1: Do this during your first time uploading the repository. What this does is create a shorthand version to make updating the dialogs easier. You only need to do this ONCE.

Line 2: Fetch, well, fetches all the new changes from the Yarn dialogs the writers are working on. They don't really have any other choice but to work on master in gist, so that's what the master is for.

Line 3: Git makes a folder called Assets/Dialogs and puts the entire git remote repository in it.

Renaming the git repository

I will need this when we actually get a name for the project LOL

Link: [How do I rename a repository on GitHub?](#)