Build Your First Twine Story

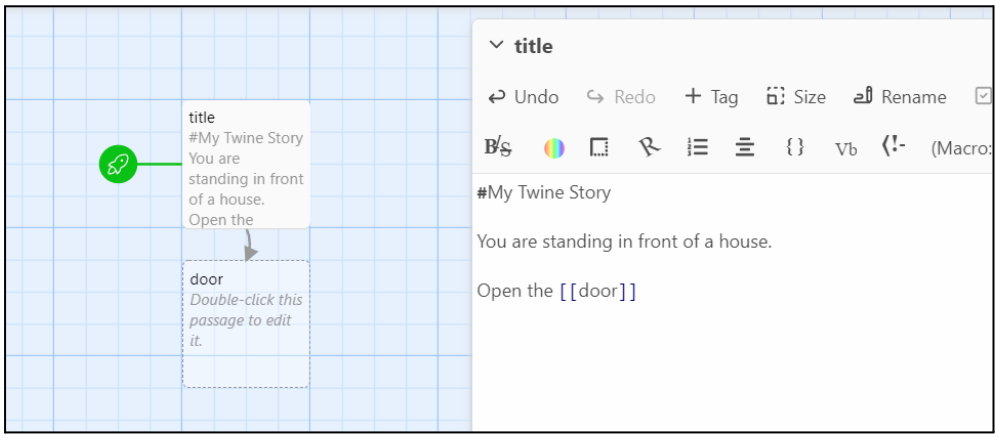
1. Click on **'+New'** under **'Story'** in the top left corner of the start screen. Rename your new story to something like 'Test' and then click **'Create'**. This will create a new story map.

2. Twine stories are made up of *passages*. These are sections of text which link together to create the story. The first passage appears automatically when you begin a new story.

Double click on the first passage to edit it. Change the name of the passage to **'title'** and type in the following text:

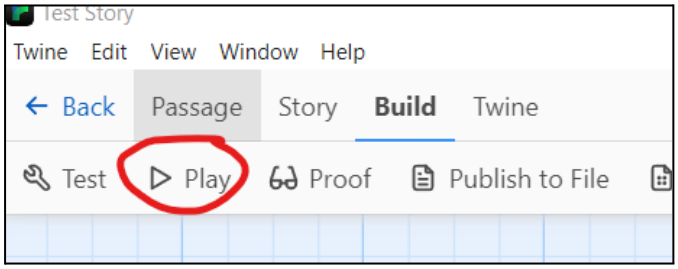
```
#My Twine Story

You are standing in front of a house.

Open the [[door]]
```

You will notice that this has created a new passage titled **'door'**. Putting double square brackets around any text will create a new passage. You can also create a passage by going to the toolbar at the top of the screen and clicking **'Passage'** and then **'+New'**.



3. At the toolbar at the top of the screen, click **'Build'** and then **'Play'** to see what your first passage looks like.

At any point while you're making a Twine story you can use the **'Play'** button to test what you've made so far. It's good practice to do this quite often so you can catch any errors quickly.



It should open a browser window showing something like this:

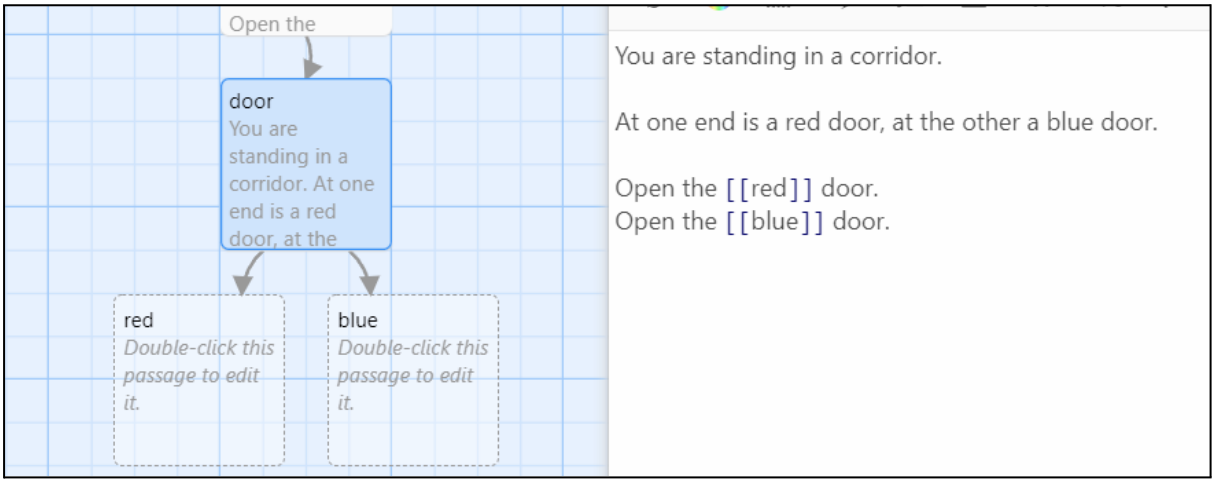
As you can see, putting **'#'** in front of the first line has changed the formatting so the text is much bigger and bolder. This is an easy way to create a header or title.

Also, putting **[[double square brackets]]** around the word 'door' has turned it into a *link*. Click through it if you like—there won't be anything there yet. This is our next job!

4. Go back to Twine and double click the passage named **'door'**. Add the following text:

```
You are standing in a corridor.

At one end is a red door, at the other a blue door.

Open the [[red]] door.
Open the [[blue]] door.
```

As you can see, this has created two new passages titled **‘red’** and **‘blue’**. This is our first *branching choice*. The player can choose either to go through the red door or the blue door.



5. Let’s add some text to these passages now. In both we want to make sure that the player has the option to return to the **‘door’** passage so they can try out a different choice.

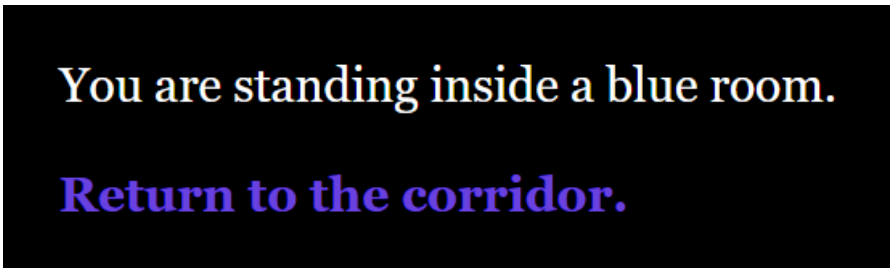
Add the following text to the **‘blue’** passage:

> You are standing inside a blue room.
>
> [[Return to the corridor.|door]]

As you can see, we’ve done something a little different with the link here. The first bit of text in the square brackets—*’Return to corridor.’*—is what the player will see on the screen. The text after the vertical line is the name of the passage it will link to—in this case, **‘door’**.

You can type a vertical line by pressing Shift+\ (the backslash key is usually to the left of the arrow keys at the bottom right of your keyboard).

Use the **‘Play’** button to see this in practice. It should look something like this:



6. Now add the following text to the **‘red’** passage:

```
You are standing inside a red room.

On the floor is a locked chest.

[[Open the chest.|chest]]
[[Return to the corridor.|door]]
```

This will create a new passage titled **‘chest’**. Open this passage and add the following text:

```
The chest is locked!

[[Go back.|red]]
```

We now have a corridor, two rooms, and a locked chest. What we need is a key—and for that we’re going to need to use a *variable*.

A *variable* is a value saved by the game, which can change depending on player choices. The options available to players might vary depending on the variables in the game.

Variables can be created or changed with a *macro*—a small piece of code—that looks like this:

```
(set: $variable to true)
```

Variables are always indicated with the **‘$’** key. The name of your variable can be made up of any combination of uppercase letters, lowercase letters and numbers, but cannot include spaces.

There are three basic kinds of variables that you might use: *strings*, *numbers*, and *booleans*.

String variables save text (words, phrases, etc.). This could be used to record things like a player’s name or their favourite colour. For example:

```
(set: $name to ‘John’)
(set: $faveColour to ‘blue’)
```

Number variables save numbers (surprise surprise). This could be used to track things like how much money a player has, or their level of skill in a certain area. For example:

```
(set: $money to 100)
(set: $strength to 6)
```

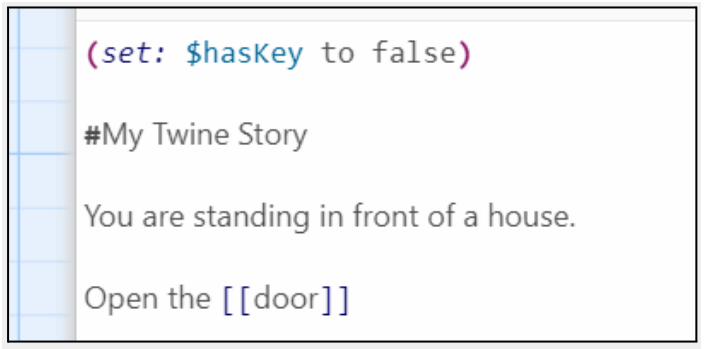
Boolean variables save true or false values. This could be used to track things like whether a player has an item in their possession, or if they have visited a certain place. For example:

```
(set: $hasGuitar to false)
(set: $beenUpstairs to true)
```

This might seem confusing if you're not used to using variables, but don't panic—we're only going to use one kind (boolean) in our test game, and we'll go through that step by step.

7. Firstly, let's create a variable to track whether the player has a key for the chest. Open the **'title'** passage again and add the following code to the first line:

(set: $hasKey to false)



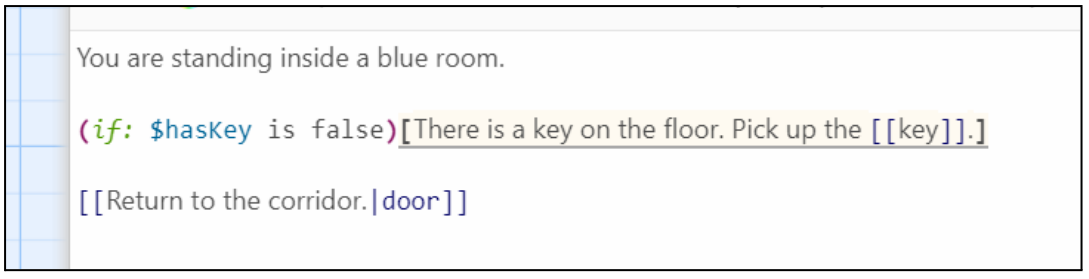
This will not be visible to the player, but it records that they do not have the key when they start the game.

8. Next, we want to put the key in the blue room. However, we only want the key to be in there if the player hasn't already picked it up. For this we'll use an *if* macro.

An *if* macro is another small piece of code that we use to check if a variable meets certain conditions. If the conditions are met, there will be a certain outcome. Let's use this key as an example.

Add the following code to the **'blue'** passage, before the link back to the corridor:

(if: $hasKey is false)[There is a key on the floor. Pick up the [[key]].]



Let's break down that code.

The first part, in the *(round brackets)* sets the conditions that need to be met—in this case, that the player doesn't have the key yet.

The second part, in the *[square brackets]*, is what the player will see if (and only if) these conditions are met. In this case, some text and a link to a new passage titled **'key'**.

9. Now we need to change that variable to reflect the player now having the key in their possession.

Add the following to the **'key'** passage:

```
(set: $hasKey to true)

You have a key!

[[Go back|blue]]
```

Use the **'Play'** button to see this all work in practice. You should notice that, after you've picked up the key, it will no longer be on the floor of the blue room when you return there.

10. So, the player now has the key—all we need to do is let them open the chest.

For this we're going to use another *if* macro, this time along with an *else* macro. These two bits of code work together to test whether a condition has been met or not. If it has, there will be one outcome; if it hasn't, there will be a different outcome.

Let's see what this looks like in practice. Delete the text in the passage titled **'chest'** and add the following:

```
(if: $hasKey is true)[
You use the key to unlock the chest. It's full of gold and riches!

#You win! Congratulations!
]
(else:)[
The chest is locked!

[[Go back.|red]]
]
```

This tests to see whether the player has the key in their possession. If they do have the key, the victory message is displayed and the game is complete. If they do not, the player fails and must go back.

11. Last but not least, go to the toolbar and click **'Build'** then **'Publish To File'** to save your finished game.

Congratulations! You have just made your first working Twine story.

The possibilities of what you can make with this software are endless. Play around with different combinations of passages, and maybe try using your own variables to make something a bit more complex. Make sure to test your work often to see how it looks in practice!