

Introduction to Twine

Using Harlowe

What is Twine?

- An open-source **interactive fiction** tool
- Programming is not required to make a great story
 - Can be enhanced with variables and logic
- Stories are automatically published to HTML, allowing it to be posted anywhere



Why learn Twine?

- Serves as a strong introduction to programming fundamentals
- Learn the basics of variables, conditionals, and functions



Getting Started

- Visit twinery.org
- Either download the twine program or use it online, any option will do
 - If you opt to download the software, follow the installation instructions and open the program

Making a new story

- Click **+Story**
- Give it a name
- Click **+Add**

0 Stories

Sort By

Edit Date

Name ^A↓ _Z

There are no stories saved in Twine right now. To get started, you can either create a new story or import an existing one from a file.




Twine

+ Story

 Import From File

 Archive

 Formats

 Language


 Help



100% space available

version 2.2.1

 [Report a bug](#)

 **Untitled
Passage**
Double-click
this passage to
edit it.

 Tutorial ▲

Quick Find



 Test

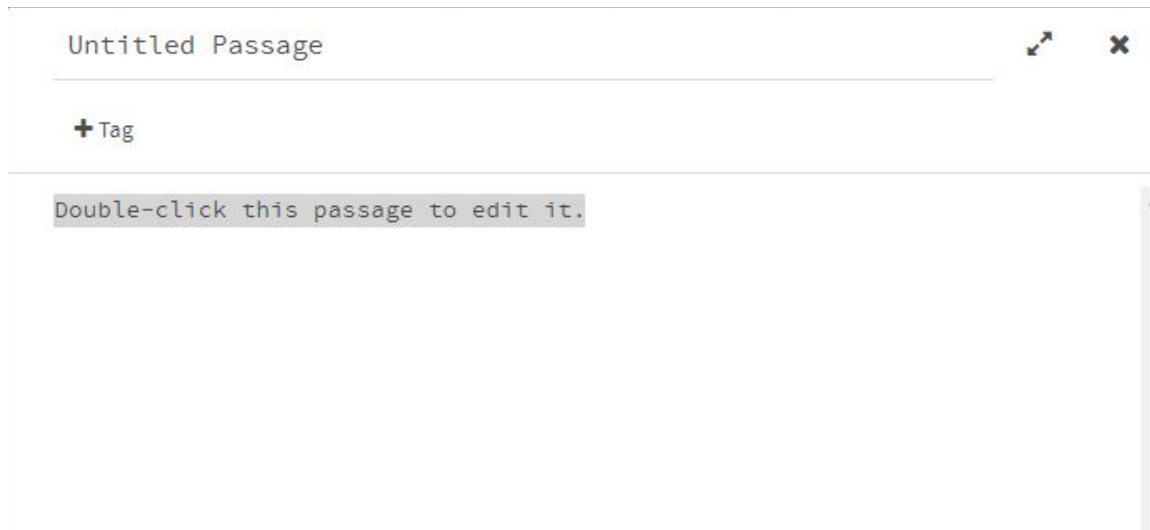
 Play

 Passage

The Interface is where you edit passages

Editing Passages

- Passages are the scenes of your story
- Describe what is going on, add dialogue, and write story elements



Connecting Passages

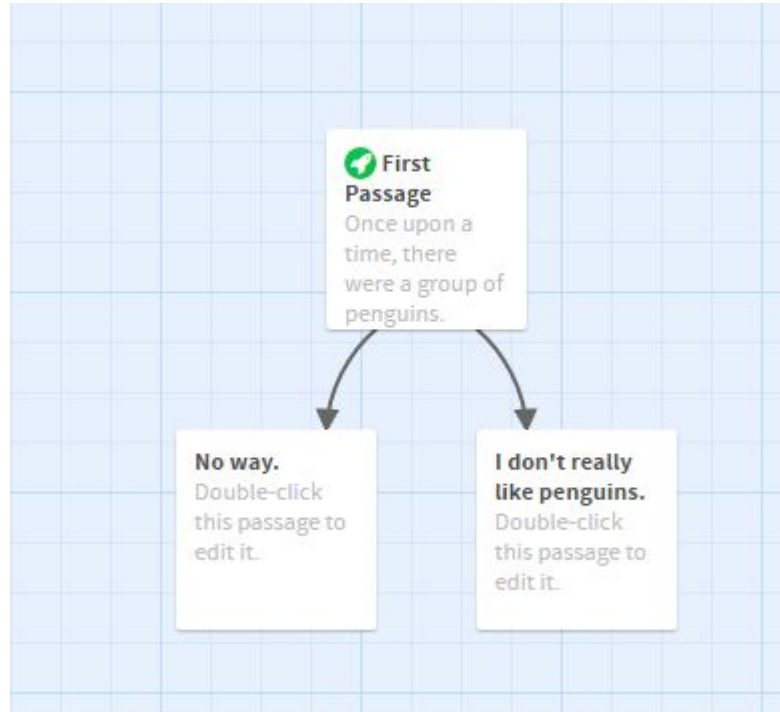
- Link passages to allow your story to branch out
- Surround text by two pairs of square brackets: `[[text]]`
- You may edit the branching passages as well

First Passage

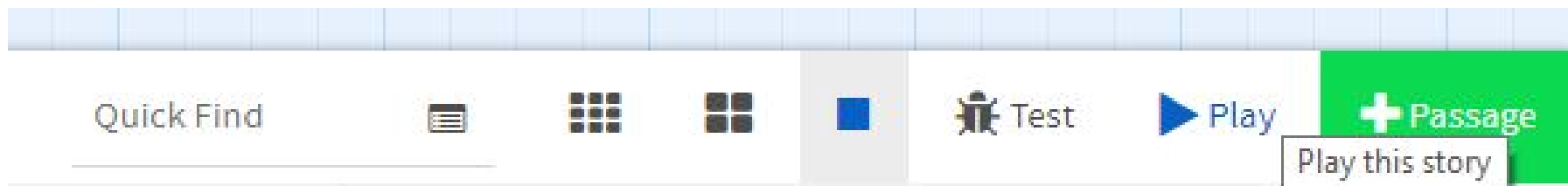


+ Tag

- Once upon a time, there were a group of penguins. These penguins had all the qualities a penguin should have: cute and cuddly.
-
- `[[No way.]]`
- `[[I don't really like penguins.]]`



Basic Twine story with branching passages



Play your story

Once upon a time, there were a group of penguins. These penguins had all the qualities a penguin should have: cute and cuddly.

No way.

I don't really like penguins.

Blue text links to another passage

Text Formatting

- You may format text differently by surrounding it with a pair of special characters

- `' 'This text is bold'`
- `//This text is italic//`
- `^^This is superscript^^`
- `---This is strikethrough---`

This text is bold

This text is italic

This is superscript

~~This is strikethrough~~

Formatted text in action

Text Formatting with HTML

- In addition to special characters, you may surround text in special HTML tags
- These tags may be used later on for CSS styling

- `<p>This is a paragraph text.</p>`
- `<i>This is bold AND italic</i>`

Programming in Twine

Variables

- Variables contain values to be used by the passages
 - They can hold strings, numbers, booleans, etc.
- Variable names start with a \$ (e.g. \$number, \$myvariable)
- Create a variable using the (set:) macro.

- `(set: Sname to "Keith Leonardo")`
- `(set: Sage to 17)`
- `Sname` is `Sage` years old.

Keith Leonardo is 17 years old.

Setting and using variables

Macros

- Perform a particular task, essentially a *function*
- Format -- (macro-name: macro-value)
- macro-values may include strings, numbers, booleans, and even the *return value* of other macros.

```
(print: “my first macro!”)
```

- `(print: "my first macro!")`

my first macro!

Visit twinery.org/wiki/harlowe:macro_markup for more macros

Hooks

- They indicate that a specific span of text is special in some way, denoted by text between square brackets [sample text]
- Hooks can be attached to macros and variables to alter the text within it.

`(text-colour: blue)[Some text]`

- `(set: $name to "Keith Leonardo")`
- `(set: $age to 17)`
- `(text-colour: blue)[$name is $age years old.]`

Keith Leonardo is 17 years old.

Hooks in action

User Input (prompt:)

- Set a variable to an input value by the user using the (prompt:) macro
- The macro takes two inputs: the prompt and default value.

```
(prompt: "What is your name?", "")
```

- `(set: Sname to (prompt: "What is your name?", ""))`
- `(print: Sname)`

What is your name?

Chiara Amisola

OK

Cancel

Chiara Amisola

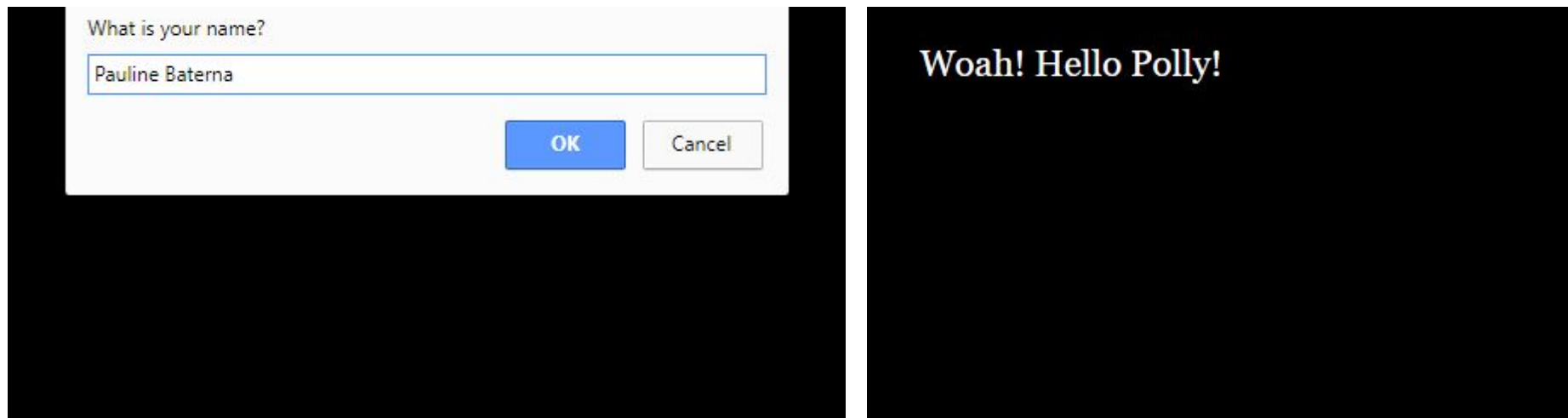
Set the user input to a variable then display it

Logical Macros

- The `(if:)`, `(else-if:)`, and `(else:)` macros allows the story to have decision making elements.
- A whole conditional block is *contained* in curly braces `{}`
- Conditionals toggle a hook's visibility depending on whether the output is `true` or `false`.

- `(set: $name to (prompt: "What is your name?", ""))`
- `Woah! {(if: $name is "Keith Leonardo") [Hello Keithy!]`
- `(else-if: $name is "Chiara Amisola") [Hello Chia!]`
- `(else-if: $name is "Pauline Baterna") [Hello Polly!]`
- `(else:) [Hello friend!]}`

A basic conditional block in action (note the curly braces)



A basic conditional block in action

Done with Twine basics!