Narrative Programming using ink

Todd Waugh Ambridge
Assistant Professor, School of Computer Science
University of Birmingham

fppt.com


What have you got?
Look what I've got.
Good-bye.
Beedle
What are you looking for?
Or are you selling?

fppt.com


THE LEGEND OF
ZELDA
BREATH OF THE WILD
Beedle
What are you looking for?
Or are you selling?
What have you got?
Look what I've got.
Good-bye.

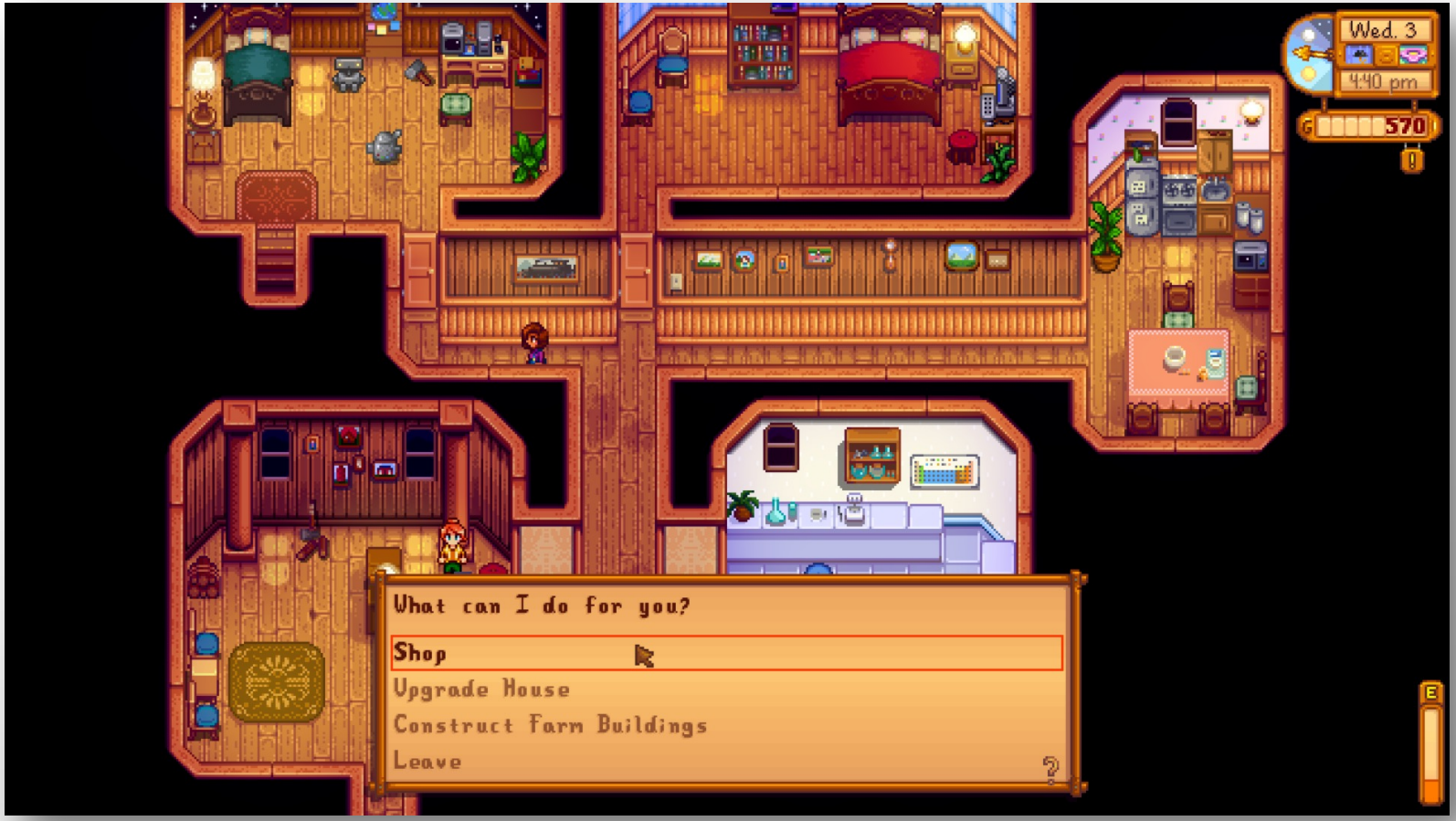
fppt.com


V
Table
1 Spurs 83
2 Chelsea 83
3 Arsenal 81
4 Liverpool 80
5 Manchester Utd 75
Final Game of the Premier League Season
Go and have some fun today
I expect nothing but a win from this match
Come on lads, show me what you can do!

fppt.com


Final Game of the Premier League Season
I expect nothing but a win from this match
Go and have some fun today
Come on lads, show me what you can do!
FIFA 20
EA SPORTS
adidas
FIFA

fppt.com


Wed. 3
4:40 pm
570
What can I do for you?
Shop
Upgrade House
Construct Farm Buildings
Leave

fppt.com


STARDEW VALLEY
Wed. 3
4:40 pm
G 570
What can I do for you?
Shop
Upgrade House
Construct Farm Buildings
Leave

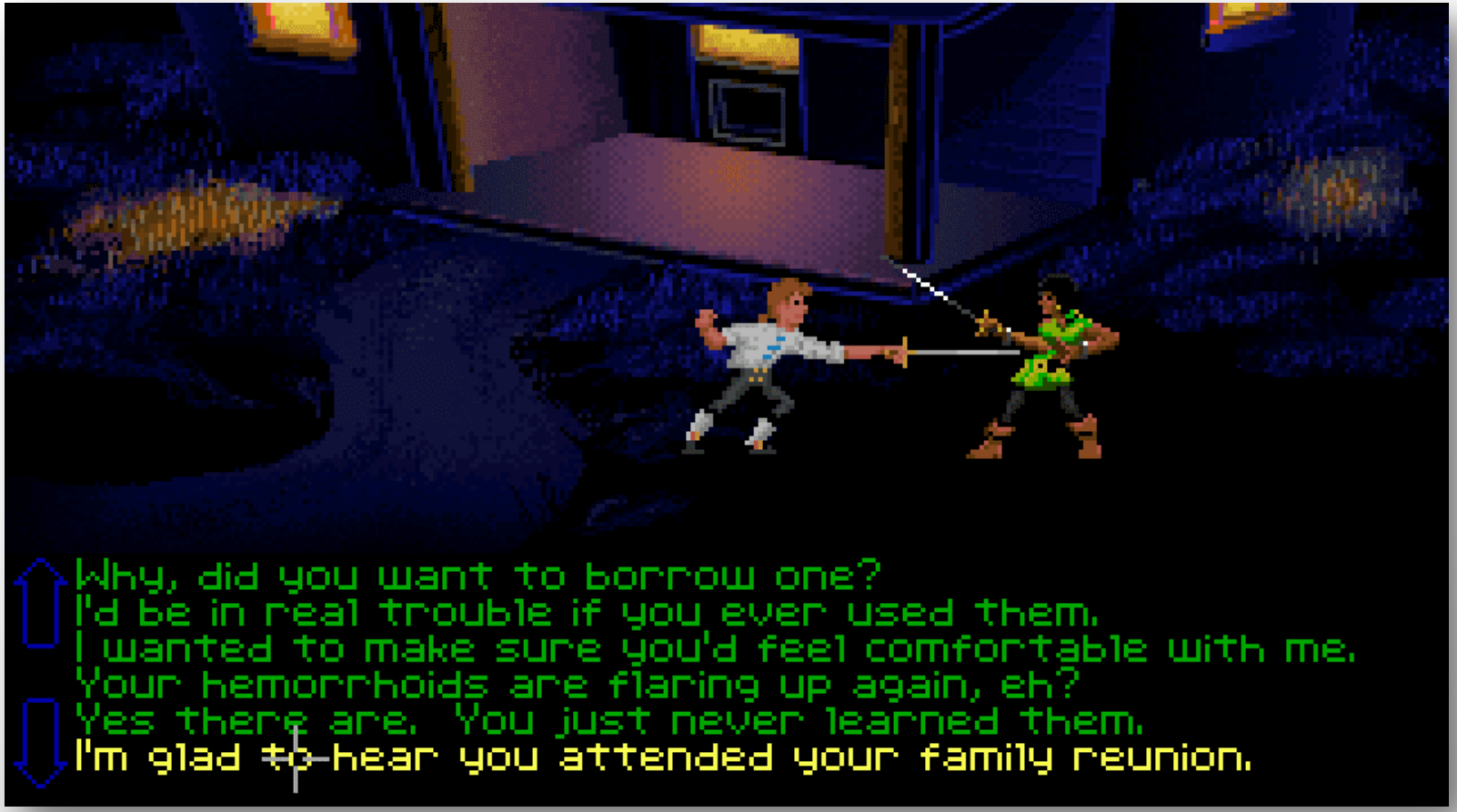
fppt.com


Narrator: *A strange symbol glows, marked on their flesh, and something within you stirs in response.*
1. Freeze and hold up your hands.
2. What happened to your friend?
3. [MEDICINE] His wounds look deep. I might be able to help.
» 4. [ILLITHID] [WISDOM] I will go where I wish.
Wisdom
Roll 1d20+2
Details:
+2 from Wisdom
Advantage/Disadvantage
Situational

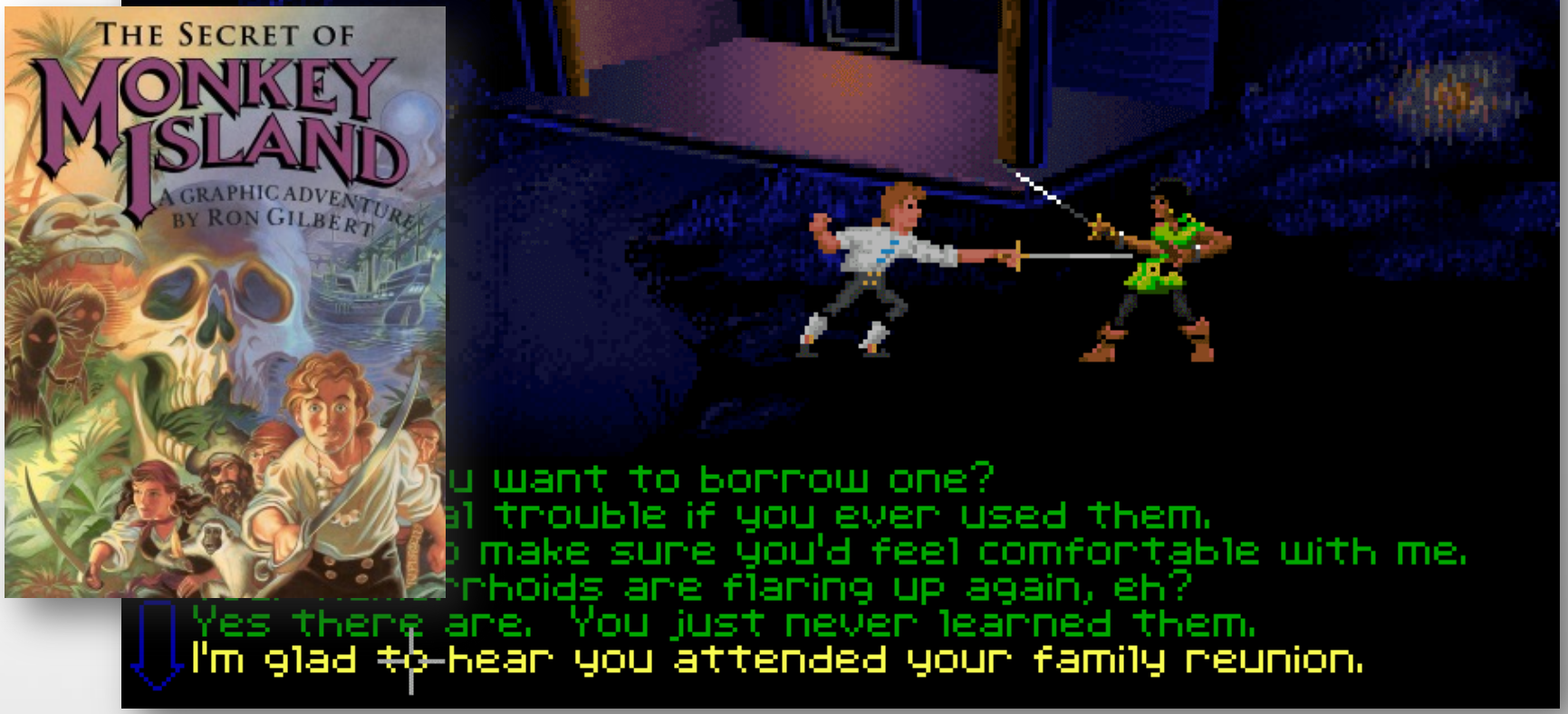
fppt.com


Baldur's Gate
Roll 1d20+2
Details:
+2 from Wisdom
Advantage/Disadvantage
Situational
Narrator: *A strange symbol glows, marked on their flesh, and something within you stirs in response.*
1. Freeze and hold up your hands.
2. What happened to your friend?
3. [MEDICINE] His wounds look deep. I might be able to help.
» 4. [ILLITHID] [WISDOM] I will go where I wish.

fppt.com

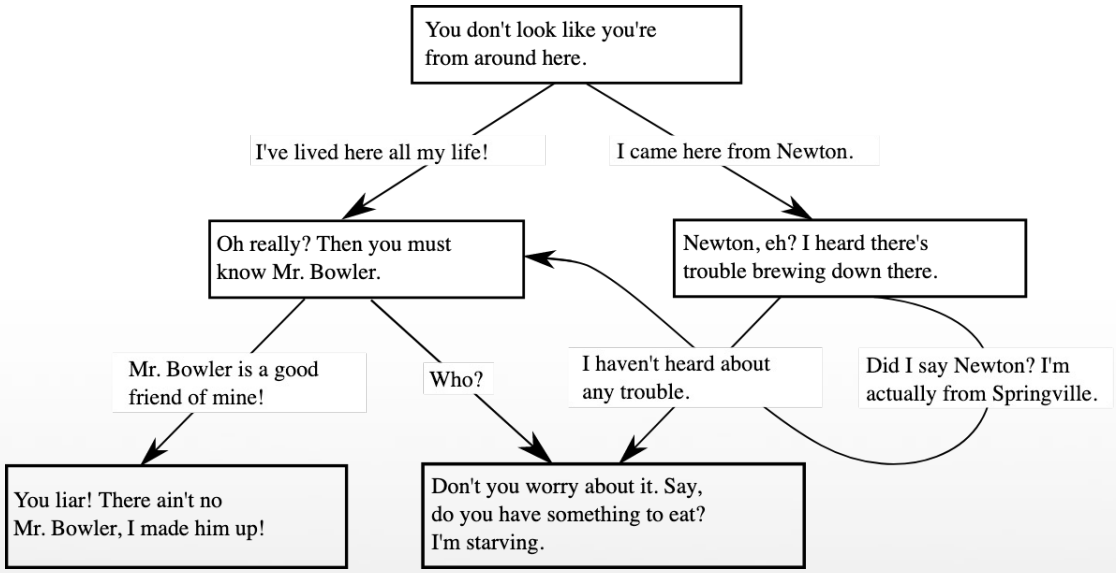

Why, did you want to borrow one?
I'd be in real trouble if you ever used them.
I wanted to make sure you'd feel comfortable with me.
Your hemorrhoids are flaring up again, eh?
Yes there are. You just never learned them.
I'm glad to hear you attended your family reunion.

fppt.com


THE SECRET OF
MONKEY ISLAND
A GRAPHIC ADVENTURE
BY RON GILBERT
u want to borrow one?
l trouble if you ever used them.
make sure you'd feel comfortable with me.
rhoids are flaring up again, eh?
Yes there are. You just never learned them.
I'm glad to hear you attended your family reunion.

fppt.com

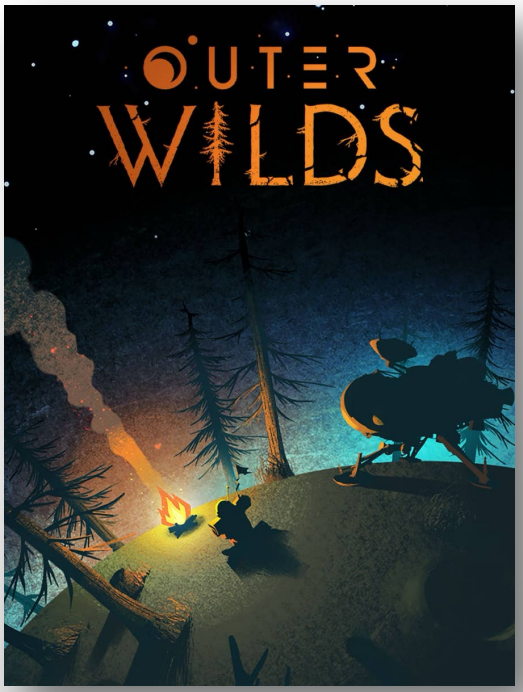
Computer games often utilise *dialogue trees*



Dialogue trees allow the player to interact with the game's *narrative*

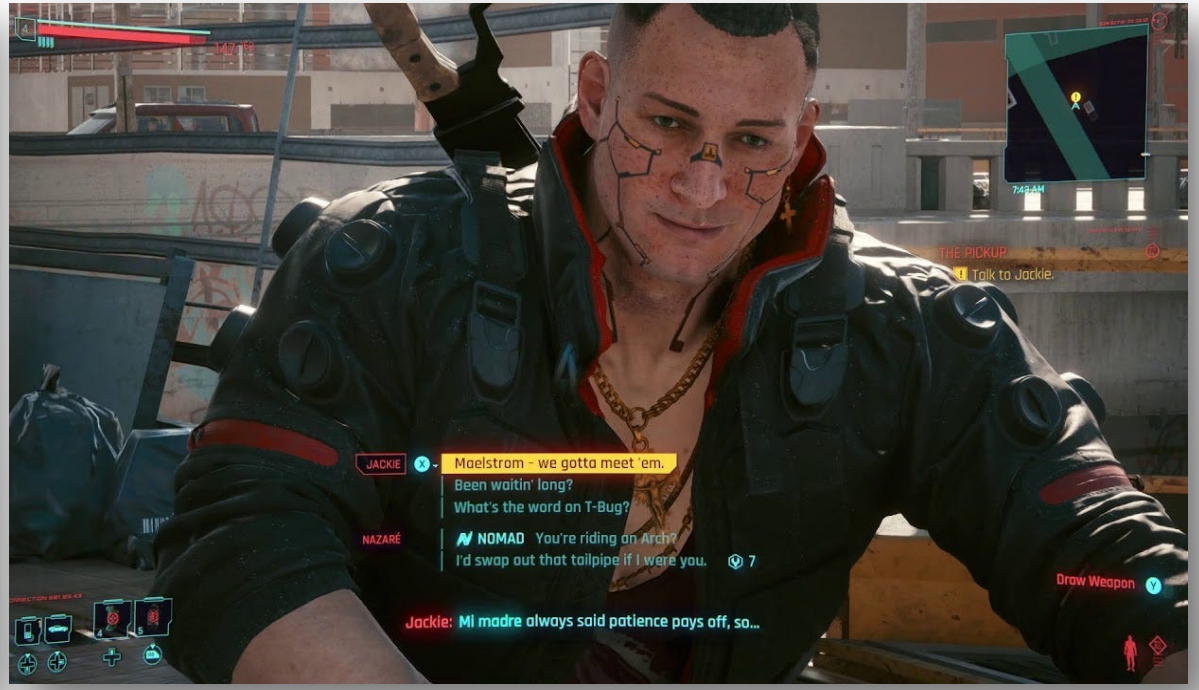
fppt.com

Some narrative games use dialogue trees only a little bit…



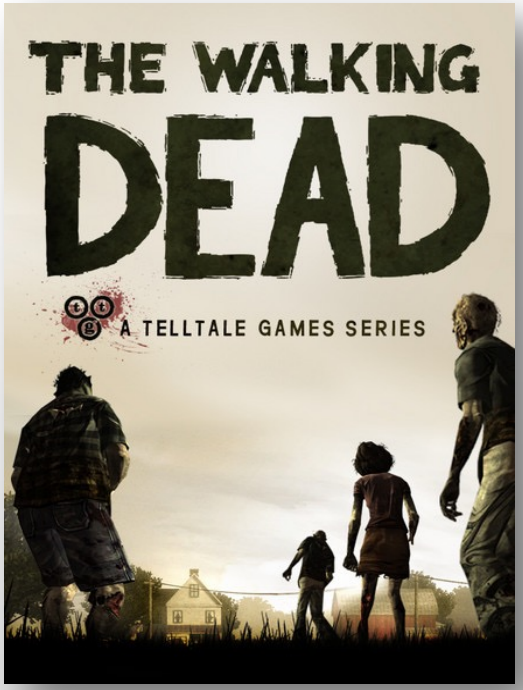
fppt.com

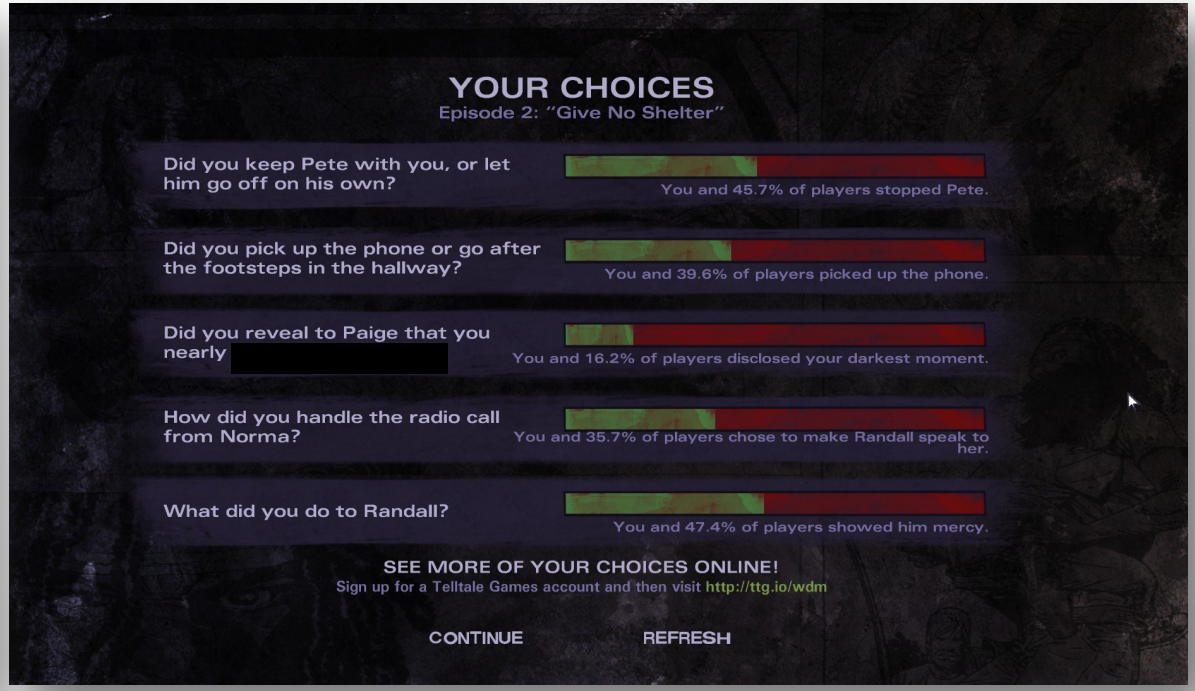
…for others it's an integral part of the game…



fppt.com

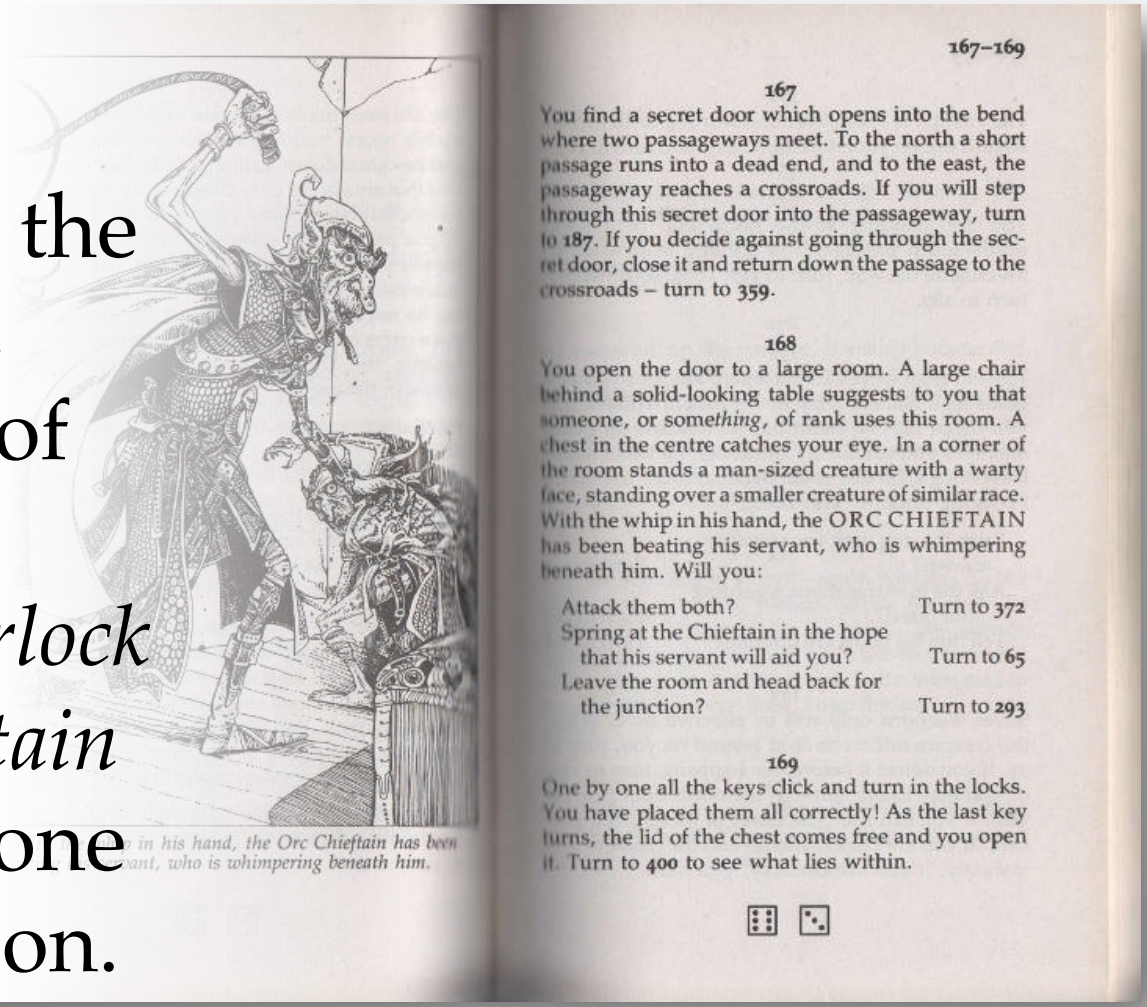
…and for a few (especially ones where *choices* matter), it's basically the entire game.





fppt.com

This style of gameplay originates from the *choose-your-own adventure books* of the early 1980s, such as *The Warlock of Firetop Mountain* by Ian Livingstone and Steve Jackson.



167

You find a secret door which opens into the bend where two passageways meet. To the north a short passage runs into a dead end, and to the east, the passageway reaches a crossroads. If you will step through this secret door into the passageway, turn to 187. If you decide against going through the secret door, close it and return down the passage to the crossroads – turn to 359.

168

You open the door to a large room. A large chair behind a solid-looking table suggests to you that someone, or something, of rank uses this room. A chest in the centre catches your eye. In a corner of the room stands a man-sized creature with a warty face, standing over a smaller creature of similar race. With the whip in his hand, the ORC CHIEFTAIN has been beating his servant, who is whimpering beneath him. Will you:

Attack them both?	Turn to 372
Spring at the Chieftain in the hope that his servant will aid you?	Turn to 65
Leave the room and head back for the junction?	Turn to 293

169

One by one all the keys click and turn in the locks. You have placed them all correctly! As the last key turns, the lid of the chest comes free and you open it. Turn to 400 to see what lies within.

in his hand, the Orc Chieftain has been ant, who is whimpering beneath him.

fppt.com

Around the same time, computerised *text adventure games* were the latest development in computer gaming, such as *Mystery House* by Roberta & Ken Williams.

fppt.com

Over time, inputting text to discover the small number of things the player could actually *do* became out-dated and was replaced with explicit choices.

fppt.com

Adventure games, and in large part the entire genre of narrative-focused games, fell out of favour in the early 2000s due to the rise of 3D and consoles.

fppt.com

But RPGs kept the torch lit, and in recent years games with a strong focus on narrative and player choice have returned to the fold.

fppt.com

Therefore, it's important for game developers to not just to think about how to program *physics* or *combat* or *puzzles*, but how to program *narrative*.

fppt.com

Therefore, it's important for game developers to not just to think about how to program *physics* or *combat* or *puzzles*, but how to program *narrative*.

In this session, that's what we'll do. And by learning about *narrative programming*, we'll understand CS more widely – as not just a technical endeavour, but a *creative* one.

fppt.com

Outline

1. What is *"narrative programming"*?
2. Introducing ***ink***
3. The Basics
4. Better Narrative Flow
5. Logic & Computation
6. Conclusion

fppt.com

Outline

1. **What is *"narrative programming"*?**
2. Introducing ***ink***
3. The Basics
4. Better Narrative Flow
5. Logic & Computation
6. Conclusion

fppt.com

What is *"narrative programming"*?

- Narrative programming is a term which here means "to program in a narrative way"

fppt.com

What is *"narrative programming"*?

- Narrative programming is a term which here means "to program in a narrative way"
  - *It's not a technical term…*

fppt.com

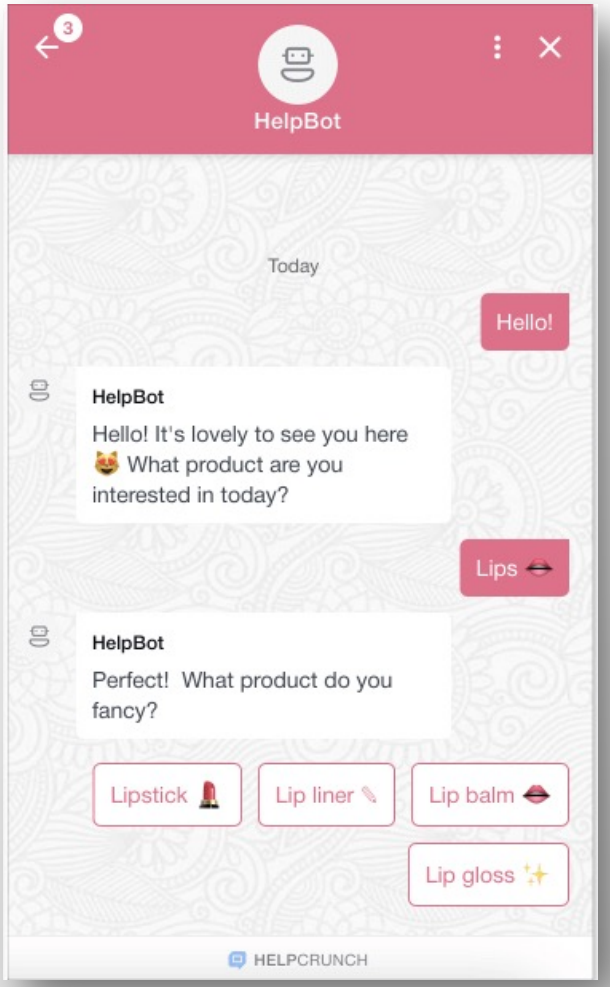
What is *"narrative programming"*?

- Narrative programming is a term which here means "to program in a narrative way"
  - *It's not a technical term…*

- As already discussed, lots of computer games have narrative features (e.g. dialogue trees) that could be written in a narrative programming language

fppt.com

- Not interested in games? Don’t worry! There are lots of other uses for narrative programming, such as chat bots that companies use to help you find the information you need.



fppt.com

Why use narrative programming?

- When we program in an *imperative* language like *Python,* the code is written as a step-by-step sequence of commands.

fppt.com

Why use narrative programming?

- When we program in an *imperative* language like *Python*, the code is written as a step-by-step sequence of commands.

```
def factorial(n):
    if n < 0:
        raise Exception("Input to factorial must be non-negative")
    else if n < 2:
        return 1
    else:
        return n * factorial(n-1)

print factorial(5)
```

fppt.com

Why use narrative programming?

- Writing a chat bot or dialogue tree in such a language is doable, but it feels clunky…

fppt.com

Why use narrative programming?

- Writing a chat bot or dialogue tree in such a language is doable, but it feels clunky…

```
match choice:
    case 0:
        print(“You walk down the north corridor.”)
        GoNorth()
    case 1:
        print(“You walk up the southern staircase.”)
        GoSouth()
    case 2:
        print(“You pick up the silver key.”)
        PickUp(“key”)
```

fppt.com

Why use narrative programming?

- But what about a narrative programming *language*?

fppt.com

Why use narrative programming?

- But what about a narrative programming *language*?
  - One whose *syntax* and *semantics* support the creation of narrative programs by design

fppt.com

Why use narrative programming?

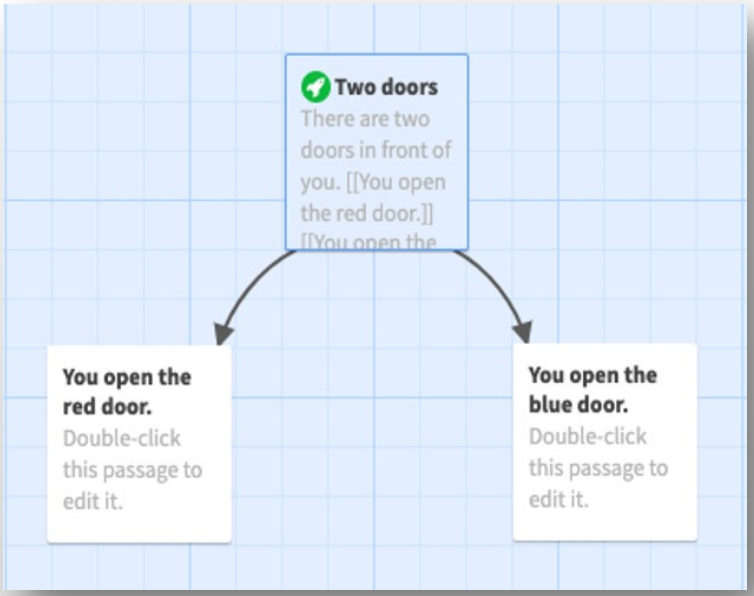
- But what about a narrative programming *language*?
  - One whose *syntax* and *semantics* support the creation of narrative programs by design

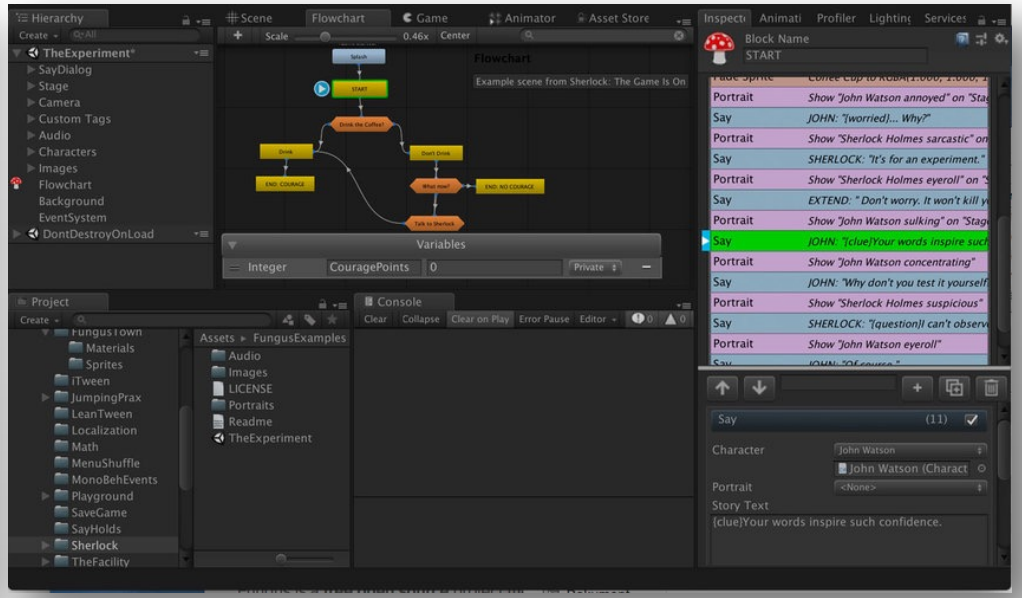
```
What do you do next?

 * I walk down the north corridor. -> GoNorth
 * I walk up the southern staircase. -> GoSouth
 * I pick up the silver key. -> PickUp(key)
```

fppt.com

Examples of narrative languages

- There are a few *graphical* tools for narrative programming, such as *Twine* and *Fungus*.





fppt.com

Examples of narrative languages

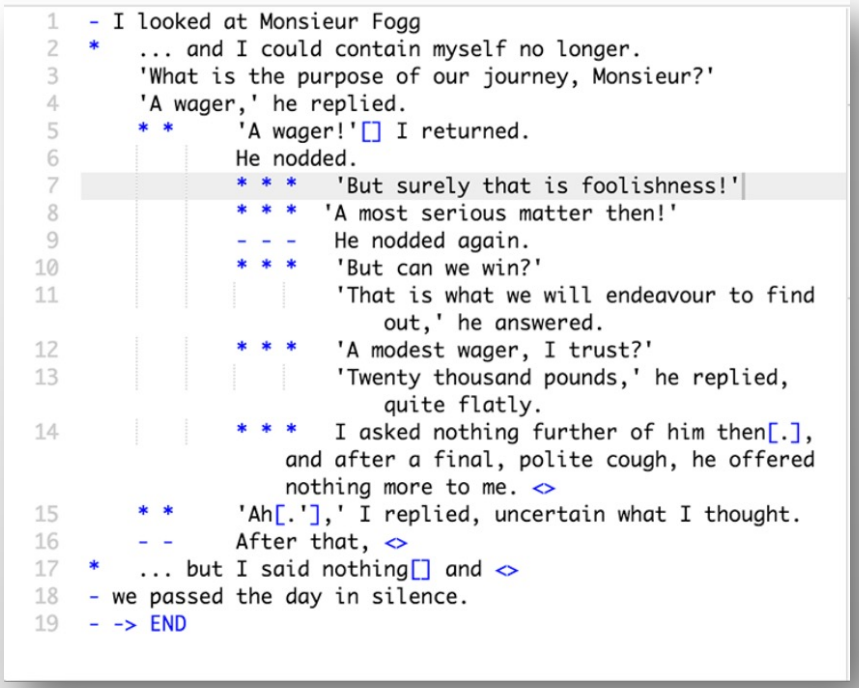
- Meanwhile, *textual* languages for narrative programming are starting to be seen

```
- I looked at Monsieur Fogg
*   ... and I could contain myself no longer.
    'What is the purpose of our journey, Monsieur?'
    'A wager,' he replied.
    * *     'A wager!'[] I returned.
            He nodded.
            * * *   'But surely that is foolishness!'
            * * *  'A most serious matter then!'
            - - -   He nodded again.
            * * *   'But can we win?'
                    'That is what we will endeavour to find
                        out,' he answered.
            * * *   'A modest wager, I trust?'
                    'Twenty thousand pounds,' he replied,
                        quite flatly.
            * * *   I asked nothing further of him then[.],
                    and after a final, polite cough, he offered
                    nothing more to me. <>
    * *     'Ah[.'],' I replied, uncertain what I thought.
    - -     After that, <>
*   ... but I said nothing[] and <>
- we passed the day in silence.
- -> END
```

fppt.com

Examples of narrative languages

- Meanwhile, *textual* languages for narrative programming are starting to be seen
  - Today, we will look at one of these and see what it can teach us about programming in a creative way



```
- I looked at Monsieur Fogg
*   ... and I could contain myself no longer.
    'What is the purpose of our journey, Monsieur?'
    'A wager,' he replied.
    * *     'A wager!'[] I returned.
            He nodded.
            * * *   'But surely that is foolishness!'
            * * *  'A most serious matter then!'
            - - -   He nodded again.
            * * *   'But can we win?'
                    'That is what we will endeavour to find
                        out,' he answered.
            * * *   'A modest wager, I trust?'
                    'Twenty thousand pounds,' he replied,
                        quite flatly.
            * * *   I asked nothing further of him then[.],
                    and after a final, polite cough, he offered
                    nothing more to me. <>
    * *     'Ah[.'],' I replied, uncertain what I thought.
    - -     After that, <>
*   ... but I said nothing[] and <>
- we passed the day in silence.
- -> END
```

fppt.com

Outline

1. What is *"narrative programming"*?
2. **Introducing *ink***
3. The Basics
4. Better Narrative Flow
5. Logic & Computation
6. Conclusion

fppt.com

Introducing *ink*

- ***ink*** is a "narrative scripting language for games" developed by Joseph Humfrey and Jon Ingold

fppt.com

Introducing *ink*

- ***ink*** is a "narrative scripting language for games" developed by Joseph Humfrey and Jon Ingold
  - Humphrey & Ignold created ***ink*** for designing their own games at their company ***inkle***

inkle

fppt.com

Introducing *ink*

- ***ink*** is used to design the narrative component (dialogue trees, choices, etc.) of ***inkle***'s games, which are made with *Unity*

fppt.com

Introducing *ink*

- But it can also be used on its own to create

fppt.com

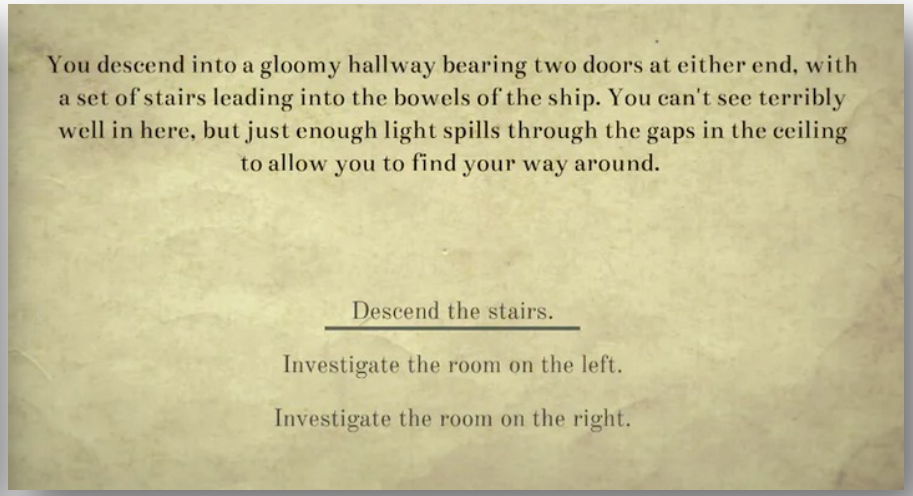
Introducing *ink*

- But it can also be used on its own to create
  - old-style text adventures,

fppt.com

Introducing *ink*

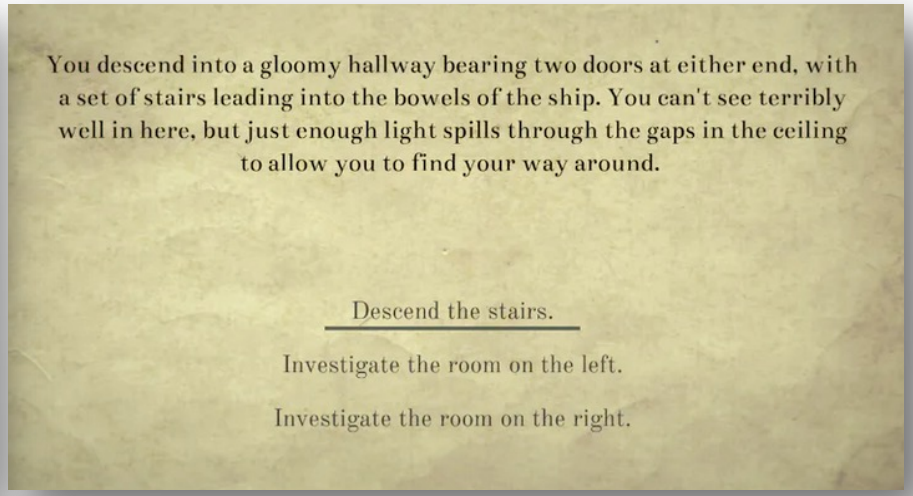
- But it can also be used on its own to create
  - old-style text adventures,
  - interactive fiction,



fppt.com

Introducing *ink*

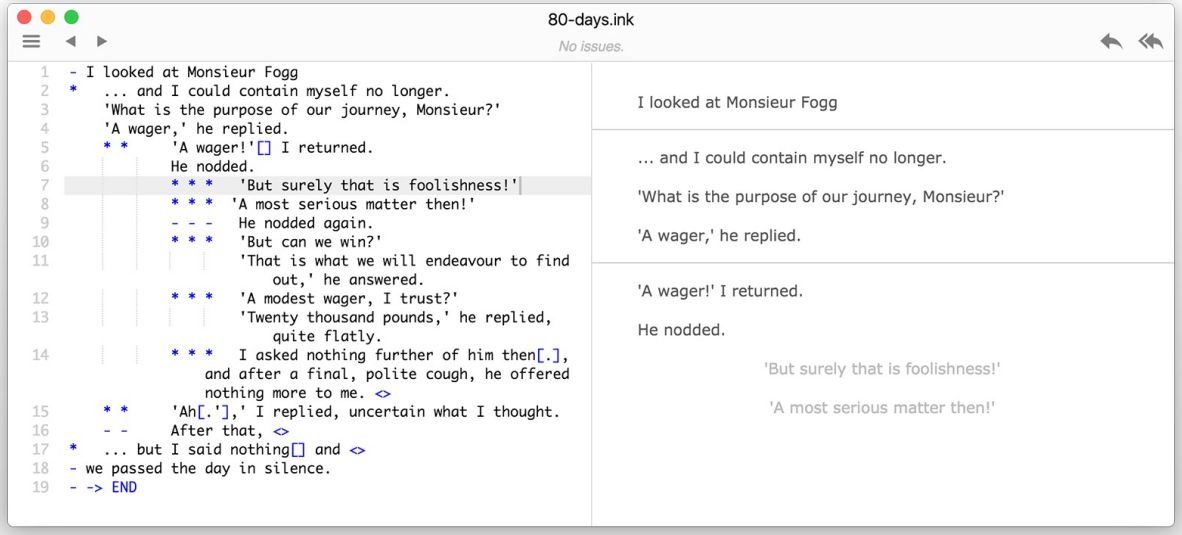
- But it can also be used on its own to create
  - old-style text adventures,
  - interactive fiction,
  - or other (non-game) programs.



fppt.com

Getting started with *inky*

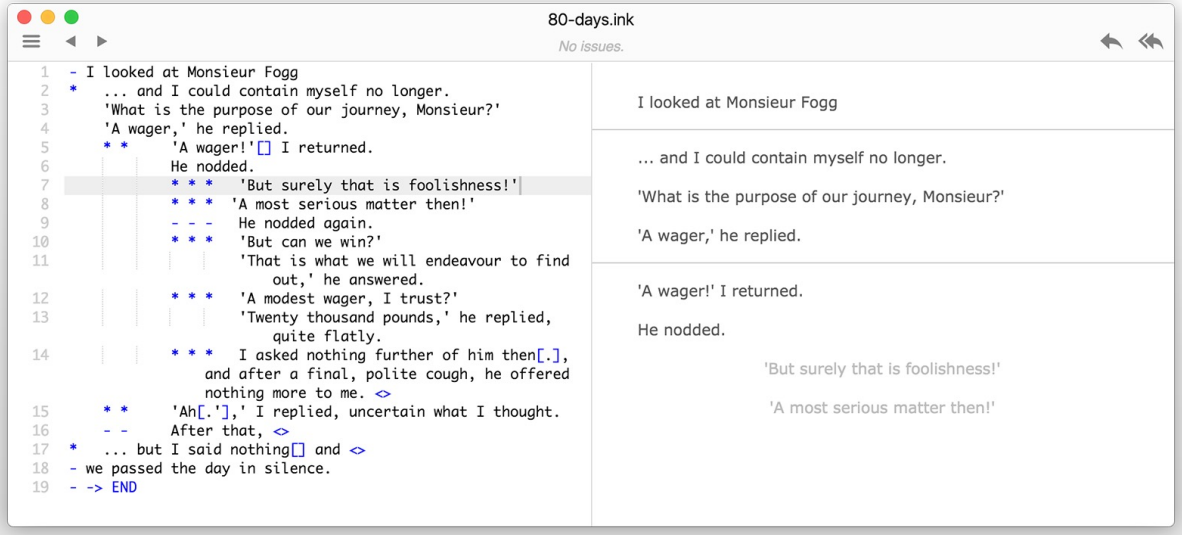
- We can write ***ink*** in any text editor and then compile it, but ***inkle*** have made a nice tool called ***inky*** that allows you to write and run ***ink*** in the same interface



fppt.com

Getting started with *inky*

- We can write ***ink*** in any text editor and then compile it, but ***inkle*** have made a nice tool called ***inky*** that allows you to write and run ***ink*** in the same interface

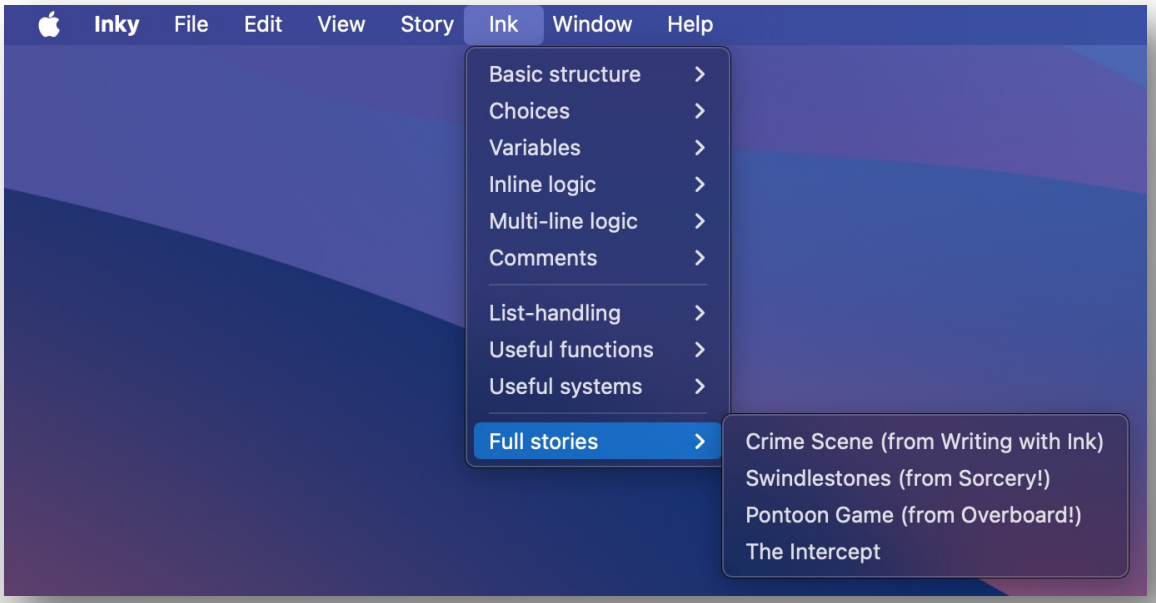


– Let's boot it up!

fppt.com

Getting started with *inky*

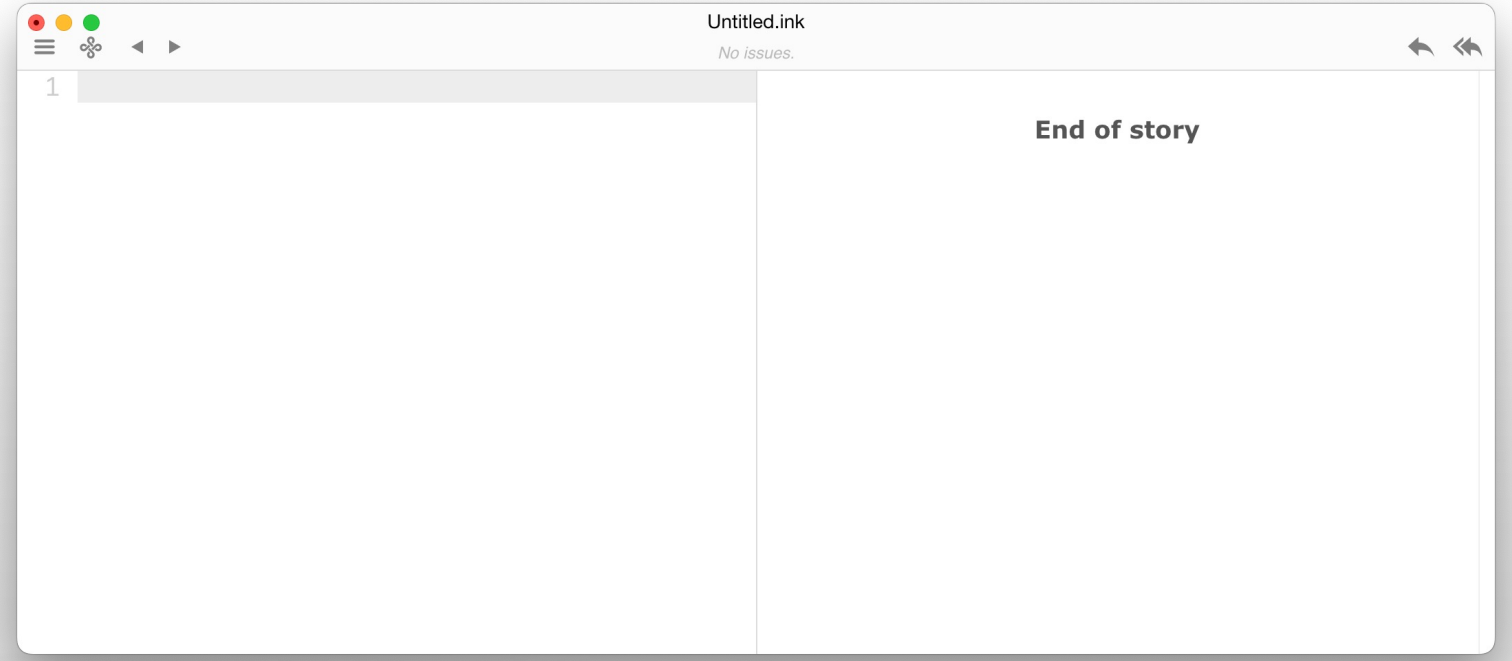
- To start with, let's play around for five minutes with an example *ink* creation
  - Choose one from *inky*'s menu of full stories



fppt.com

Getting started with *inky*

- Now, let's learn how to write some ***ink***
  - Create a *New Project* and remove any text that's already there



fppt.com

Outline

1. What is *"narrative programming"*?
2. Introducing ***ink***
3. **The Basics**
4. Better Narrative Flow
5. Logic & Computation
6. Conclusion

fppt.com

The Basics

- The basic concepts for designing narrative flow in ***ink*** are

fppt.com

The Basics

- The basic concepts for designing narrative flow in ***ink*** are
  - text,

fppt.com

The Basics

- The basic concepts for designing narrative flow in ***ink*** are
  - text,
  - choices,

fppt.com

The Basics

- The basic concepts for designing narrative flow in ***ink*** are
  - text,
  - choices,
  - knots, diverts and branching.

fppt.com

Text

- The simplest ***ink*** script is just some *text*

fppt.com

Text

- The simplest ***ink*** script is just some *text*

```
Tick tock.
Tick tock.
Tick tock.
You can’t sleep. You’ve been trying for hours and nothing has
worked.
```

fppt.com

Text

- The simplest ***ink*** script is just some *text*
  - ***inky*** will run this script: it shows us the text and immediately ends the story afterwards

```
Tick tock.
Tick tock.
Tick tock.
You can’t sleep. You’ve been trying for hours and nothing has worked.
```

fppt.com

Text

- The simplest ***ink*** script is just some *text*
  - ***inky*** will run this script: it shows us the text and immediately ends the story afterwards
  - We can also add comments with two slashes //

```
// Realistic to have an analogue clock nowadays? Maybe remove.
Tick tock.
Tick tock.
Tick tock.
You can’t sleep. You’ve been trying for hours and nothing has worked.
```

fppt.com

Choices

- The power of directing narrative flow comes from *choices*

```
What would you like to do?
```

fppt.com

Choices

- The power of directing narrative flow comes from *choices*

```
What would you like to do?

 * Try to count sheep.
```

fppt.com

Choices

- The power of directing narrative flow comes from *choices*

```
What would you like to do?

 * Try to count sheep.
 * Turn your pillow over to the cool side.
```

fppt.com

Choices

- The power of directing narrative flow comes from *choices*

```
What would you like to do?

 * Try to count sheep.
 * Turn your pillow over to the cool side.
 * Get out of bed.
```

fppt.com

Choices

- The power of directing narrative flow comes from *choices*
  - Choices are denoted by asterisks *

```
What would you like to do?

 * Try to count sheep.
 * Turn your pillow over to the cool side.
 * Get out of bed.
```

fppt.com

Choices – Prompt Text

- Square brackets [ ] denote the *prompt text*

```
What would you like to do?

 * Try to count sheep.
 * Turn your pillow over to the cool side.
 * [Get out of bed.]
```

fppt.com

Choices – Prompt Text

- Square brackets [ ] denote the *prompt text*
  - Anything after] won't be part of the prompt

```
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
 * Turn your pillow over to the cool side.
 * [Get out of bed.]
```

fppt.com

Choices – Prompt Text

- Square brackets [ ] denote the *prompt text*
  - Anything after] won't be part of the prompt
  - Furthermore, anything written in [ ] won't be printed after the choice is made

```
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
 * Turn your pillow over[ to the cool side.]? You just did that!
 * [Get out of bed.]
```

fppt.com

Choices – Nested Choices

- Multiple nested asterisks * can be used to create nested choices

```
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
```

fppt.com

Choices – Nested Choices

- Multiple nested asterisks * can be used to create nested choices

```
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
 * * One...
```

fppt.com

Choices – Nested Choices

- Multiple nested asterisks * can be used to create nested choices

```
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
 * * One...
 * * * Two...
```

fppt.com

Choices – Nested Choices

- Multiple nested asterisks * can be used to create nested choices

```
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
 * * One...
 * * * Two...
 * * * * Three...

 * * * * Four... [] Wait, that's not right.
```

fppt.com

Choices – Nested Choices

- Multiple nested asterisks * can be used to create nested choices
  - Though this is quickly getting confusing…

```
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
 * * One...
 * * * Two...
 * * * * Three...
 * * * * * Four...
 * * * * Four... [] Wait, that's not right.
```

fppt.com

Knots

- Labelled sections of code are called *knots*
  - These are like functions or subroutines in imperative languages like *Python*
  - They are denoted with `=== knot_name ===`

```
What would you like to do?
 * [Try to count sheep.]
// Rest of first choice code goes here


You decide to try to count sheep...
// Rest of counting sheep code goes here
```

fppt.com

Knots

- Labelled sections of code are called *knots*
  - These are like functions or subroutines in imperative languages like *Python*
  - They are denoted with `=== knot_name ===`

```
=== first_choice ===
What would you like to do?
 * [Try to count sheep.]
// Rest of first choice code goes here

=== count_sheep ===
You decide to try to count sheep...
// Rest of counting sheep code goes here
```

fppt.com

Diverts

- These knots have broken our narrative flow ☹

```
You can’t sleep. You’ve been trying for hours and nothing has
worked.

=== first_choice ===
What would you like to do?
 * [Try to count sheep.]
// Rest of first choice code goes here
```

fppt.com

Diverts

- These knots have broken our narrative flow ☹
  - Knots aren't run by default!

```
You can't sleep. You've been trying for hours and nothing has
worked.

=== first_choice ===
What would you like to do?
 * [Try to count sheep.]
// Rest of first choice code goes here
```

fppt.com

Diverts

- These knots have broken our narrative flow ☹
  - Knots aren't run by default!
  - So, you need to *divert* to them with arrows `->`

```
You can't sleep. You've been trying for hours and nothing has
worked. -> first_choice

=== first_choice ===
What would you like to do?
 * [Try to count sheep.]
// Rest of first choice code goes here
```

fppt.com

Diverts – Branching

- Knots and diverts help to tidy up your ***ink***

```
=== first_choice ===
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...

 * Turn your pillow over[ to the cool side.]? You just did that!

 * [Get out of bed.]
```

fppt.com

Diverts – Branching

- Knots and diverts help to tidy up your ***ink***
  - Diverting within multiple choices is called *branching*

```
=== first_choice ===
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
   -> count_sheep
 * Turn your pillow over[ to the cool side.]? You just did that!
   -> first_choice
 * [Get out of bed.]
```

fppt.com

Diverts – Branching

- Knots and diverts help to tidy up your ***ink***
  - Diverting within multiple choices is called *branching*
  - You also need to make sure you divert to END

```
=== first_choice ===
What would you like to do?

 * [Try to count sheep.] You decide to try to count sheep...
   -> count_sheep
 * Turn your pillow over[ to the cool side.]? You just did that!
   -> first_choice
 * [Get out of bed.] -> END
```

fppt.com

```
// Realistic to have an analogue clock nowadays? Maybe remove.
Tick tock.
Tick tock.
Tick tock.
You can’t sleep. You’ve been trying for hours and nothing has
worked. -> first_choice

=== first_choice ===
What would you like to do?
 * [Try to count sheep.] You decide to try to count sheep...
   -> count_sheep
 * Turn your pillow over[ to the cool side.]? You just did that!
   -> first_choice
 * [Get out of bed.] -> END

=== count_sheep ===
You decide to try to count sheep...
 * One...
 * * Two...
 * * * Three...
 * * * * Four... -> END
 * * * Four... [] Wait, that's not right. -> first_choice
```

fppt.com

Outline

1. What is *"narrative programming"*?
2. Introducing ***ink***
3. The Basics
4. **Better Narrative Flow**
5. Logic & Computation
6. Conclusion

fppt.com

Better Narrative Flow

- The more advanced concepts for designing better narrative flow in ***ink*** are

fppt.com

Better Narrative Flow

- The more advanced concepts for designing better narrative flow in ***ink*** are
  - sticky, default and conditional choices,

fppt.com

Better Narrative Flow

- The more advanced concepts for designing better narrative flow in ***ink*** are
  - sticky, default and conditional choices,
  - tunnels and gathers,

fppt.com

Better Narrative Flow

- The more advanced concepts for designing better narrative flow in ***ink*** are
  - sticky, default and conditional choices,
  - tunnels and gathers,
  - conditional and variable text.

fppt.com

Special Choices

- As we've seen, we can divert back to the same knot, creating a sort-of *while loop*

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the past
eight months. I'm going to...

 * [...stand up.] I'm going to stand up. -> cell
 * [...recall why I'm here.] I'm going to recall why I'm here.
   -> recall
 * [...wait a little longer.] -> bench
```

fppt.com

Special Choices

- As we've seen, we can divert back to the same knot, creating a sort-of *while loop*
  - Each choice only appears once in the loop: we can't take the same path twice

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the past
eight months. I'm going to...

 * [...stand up.] I'm going to stand up. -> cell
 * [...recall why I'm here.] I'm going to recall why I'm here.
   -> recall
 * [...wait a little longer.] -> bench
```

fppt.com

Special Choices – Sticky

- But what if we want to?

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the past
eight months. I'm going to...

 * [...stand up.] I'm going to stand up. -> cell
 * [...recall why I’m here.] I'm going to recall why I’m here.
   -> recall
 * [...wait a little longer.] -> bench
```

fppt.com

Special Choices – Sticky

- But what if we want to?
  - Simple: We replace the asterisk * with a plus + to make the choice 'sticky'

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the past
eight months. I'm going to...

 * [...stand up.] I'm going to stand up. -> cell
 * [...recall why I'm here.] I'm going to recall why I'm here.
   -> recall
 + [...wait a little longer.] -> bench
```

fppt.com

Special Choices – Sticky

- But what if we want to?
  - Simple: We replace the asterisk * with a plus + to make the choice 'sticky'
  - It will now stick around if we revisit this knot

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the past
eight months. I'm going to...

 * [...stand up.] I'm going to stand up. -> cell
 * [...recall why I'm here.] I'm going to recall why I'm here.
   -> recall
 + [...wait a little longer.] -> bench
```

fppt.com

Special Choices – Default

- Similarly, if we have a loop we might want to keep looping until we run out of options

```
=== recall ===
I was framed for a crime I didn't commit. A...

 * [...fraud scheme.] No, that’s not right... -> recall
 * [...bank robbery.] No, that’s wrong too... -> recall
```

fppt.com

Special Choices – Default

- Similarly, if we have a loop we might want to keep looping until we run out of options
  - To achieve this we use default choices

```
=== recall ===
I was framed for a crime I didn't commit. A...

 * [...fraud scheme.] No, that’s not right... -> recall
 * [...bank robbery.] No, that’s wrong too... -> recall
```

fppt.com

Special Choices – Default

- Similarly, if we have a loop we might want to keep looping until we run out of options
  - To achieve this we use default choices
  - These are marked with `*  ->` and then a newline

```
=== recall ===
I was framed for a crime I didn't commit. A...

 * [...fraud scheme.] No, that's not right... -> recall
 * [...bank robbery.] No, that's wrong too... -> recall
 * ->
   A murder. A cold-blooded murder. -> bench
```

fppt.com

Special Choices – Conditional

- We might also want certain choices to only appear under certain conditions

```
=== cell ===
I couldn't believe my luck. The key was in reach! I'll...

 * [...sit back down.] I'll sit back down. -> bench
 * [...recall why I'm here.]
   I'll recall why I'm here. -> recall
 * [...grab the key.] I'll grab the key. -> key
 * [...unlock the jail cell.]
   I'll unlock the jail cell. -> unlock
```

fppt.com

Special Choices – Conditional

- We might also want certain choices to only appear under certain conditions
  - For example, a choice should appear only if we've made a different choice

```
=== cell ===
I couldn't believe my luck. The key was in reach! I'll...

 * [...sit back down.] I'll sit back down. -> bench
 * [...recall why I'm here.]
   I'll recall why I'm here. -> recall
 * [...grab the key.] I'll grab the key. -> key
 * [...unlock the jail cell.]
   I'll unlock the jail cell. -> unlock
```

fppt.com

Special Choices – Conditional

- Conditional choices are achieved by writing the condition in squiggly brackets {} before the choice

```
=== cell ===
I couldn't believe my luck. The key was in reach! I'll...

 * [...sit back down.] I'll sit back down. -> bench
 * [...recall why I'm here.]
   I'll recall why I'm here. -> recall
 * [...grab the key.] I'll grab the key. -> key
 * {key} [...unlock the jail cell.]
   I'll unlock the jail cell. -> unlock
```

fppt.com

Special Choices – Conditional

- Conditional choices are achieved by writing the condition in squiggly brackets {} before the choice
  - We can also use not to do the opposite…

```
=== cell ===
I couldn't believe my luck. The key was in reach! I'll...

 * [...sit back down.] I'll sit back down. -> bench
 * {not recall} [...recall why I'm here.]
   I'll recall why I'm here. -> recall
 * [...grab the key.] I'll grab the key. -> key
 * {key} [...unlock the jail cell.]
   I'll unlock the jail cell. -> unlock
```

fppt.com

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the
past eight months. I'm going to...
 * [...stand up.] I'm going to stand up. -> cell
 * {not recall} [...recall why I'm here.]
   I'm going to recall why I'm here. -> recall
 + [...wait a little longer.] -> bench

=== recall ===
I was framed for a crime I didn't commit. A...
 * [...fraud scheme.] No, that's not right... -> recall
 * [...bank robbery.] No, that's wrong too... -> recall
 * ->
   A murder. A cold-blooded murder. -> bench

=== cell ===
I couldn't believe my luck. The key was in reach! I'll...
 * [...sit back down.] I'll sit back down. -> bench
 * {not recall} [...recall why I'm here.]
   I'll recall why I'm here. -> recall
 * [...grab the key.] I'll grab the key. -> key
 * {key} [...unlock the jail cell.]
   I'll unlock the jail cell. -> unlock
```

fppt.com

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the
past eight months. I'm going to...
 * [...stand up.] I'm going to stand up. -> cell
 * {not recall} [...recall why I'm here.]
   I'm going to recall why I'm here. -> recall
 + [...wait a little longer.] -> bench

=== recall ===
I was framed for a crime I didn't commit. A...
 * [...fraud scheme.] No, that's not right... -> recall
 * [...bank robbery.] No, that's wrong too... -> recall
 * ->
   A murder. A cold-blooded murder. -> bench

=== cell ===
I couldn't believe my luck. The key was in reach! I'll...
 * [...sit back down.] I'll sit back down. -> bench
 * {not recall} [...recall why I'm here.]
   I'll recall why I'm here. -> recall
 * [...grab the key.] I'll grab the key. -> key
 * {key} [...unlock the jail cell.]
   I'll unlock the jail cell. -> unlock
```

fppt.com

Weaving the Flow – Tunnels

- Sometimes we want to divert back to wherever we came from

```
 I'm going to recall why I’m here. -> recall
 * [...wait a little longer.] -> bench

=== recall ===
I was framed for a crime I didn't commit. A...
 * [...fraud scheme.] No, that’s not right... -> recall
 * [...bank robbery.] No, that’s wrong too... -> recall
 * ->
   A murder. A cold-blooded murder. -> bench
```

fppt.com

Weaving the Flow – Tunnels

- Sometimes we want to divert back to wherever we came from
  - For this, we write a tunnel `-> first -> second`

```
 I'm going to recall why I'm here. -> recall -> bench
 * [...wait a little longer.] -> bench

=== recall ===
I was framed for a crime I didn't commit. A...
 * [...fraud scheme.] No, that's not right... -> recall
 * [...bank robbery.] No, that's wrong too... -> recall
 * ->
   A murder. A cold-blooded murder. -> bench
```

fppt.com

Weaving the Flow – Tunnels

- Sometimes we want to divert back to wherever we came from
  - For this, we write a tunnel `-> first -> second`
  - We return from a tunnel with `->->`

```
I'm going to recall why I'm here. -> recall -> bench
* [...wait a little longer.] -> bench

=== recall ===
I was framed for a crime I didn't commit. A...
* [...fraud scheme.] No, that's not right... -> recall
* [...bank robbery.] No, that's wrong too... -> recall
* ->
  A murder. A cold-blooded murder. ->->
```

fppt.com

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the
past eight months. I'm going to...
 * [...stand up.] I'm going to stand up. -> cell
 * {not recall} [...recall why I'm here.]
   I'm going to recall why I'm here. -> recall -> bench
 + [...wait a little longer.] -> bench

=== recall ===
I was framed for a crime I didn't commit. A...
 * [...fraud scheme.] No, that's not right... -> recall
 * [...bank robbery.] No, that's wrong too... -> recall
 * ->
   A murder. A cold-blooded murder ->->

=== cell ===
I couldn't believe my luck. The key was in reach! I'll...
 * [...sit back down.] I'll sit back down. -> bench
 * {not recall} [...recall why I'm here.]
   I'll recall why I'm here. -> recall -> cell
 * [...grab the key.] I'll grab the key. -> key
 * {key} [...unlock the jail cell.]
   I'll unlock the jail cell. -> unlock
```

fppt.com

Weaving the Flow – Gathers

- Other times, we want the narrative to press on no matter which choice we take

```
=== key ===
Crouching down, gazing at the conveniently-placed key, I noticed it
was...
  * ...shiny. [] It glistened in the pale moonlight of the cell.
  * ...elegantly cut. [] Its grooves beckoned to be used to purchase
my freedom.
```

fppt.com

Weaving the Flow – Gathers

- Other times, we want the narrative to press on no matter which choice we take
  - These 'false choices' can be *gathered* together with a hyphen -

```
=== key ===
Crouching down, gazing at the conveniently-placed key, I noticed it
was...
 * ...shiny. [] It glistened in the pale moonlight of the cell.
 * ...elegantly cut. [] Its grooves beckoned to be used to purchase
my freedom.
 - I picked it up and got back on my feet. -> cell
```

fppt.com

Special Text

- The text so far in our creations is very static
  - This is especially true if we revisit the same knot
  - So let's make our text for dynamic!

```
=== cell ===
I couldn't believe my luck. The key was in reach! I'll...
// Rest of standing in cell code goes here
```

fppt.com

Special Text – Conditional

- First, let's add *conditional* text

```
=== cell ===
I couldn't believe my luck. The key was in reach! I'll...
// Rest of standing in cell code goes here
```

fppt.com

Special Text – Conditional

- First, let's add *conditional* text
  - This is similar to conditional choices

```
=== cell ===
I couldn't believe my luck.
{not key: The key was in reach!}
{key: I now held the key in my hand.}
I'll...
// Rest of standing in cell code goes here
```

fppt.com

Special Text – Conditional

- First, let's add *conditional* text
  - This is similar to conditional choices
  - *Top tip!* `<>` is used to stitch together text into one paragraph

```
=== cell ===
I couldn't believe my luck. <>
{not key: The key was in reach!}
{key: I now held the key in my hand.}
<> I'll...
// Rest of standing in cell code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text

```
=== bench ===
I'm sitting on the bench of my jail cell; my home for the past
eight months. I'm going to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time

```
=== bench ===
I'm sitting on the bench of my jail cell; my
{home|prison|purgatory} for the past eight months. I'm going to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time
  - Normal sequences get stuck on the *last* entry

```
=== bench ===
I'm sitting on the bench of my jail cell; my
{home|prison|purgatory} for the past eight months. I'm going to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time

```
=== bench ===
I'm {&sitting|lying} on the bench of my jail cell; my
{home|prison|purgatory} for the past eight months. I'm going to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time
  - And & sequences *cycle* their entries

```
=== bench ===
I'm {&sitting|lying} on the bench of my jail cell; my
{home|prison|purgatory} for the past eight months. I'm going to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time

```
=== bench ===
I'm {&sitting|lying} on the bench of my jail cell; my
{home|prison|purgatory}{! for the past eight months| these days}.
I'm going to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time
  - Mark ! sequences show each entry *only once*

```
=== bench ===
I'm {&sitting|lying} on the bench of my jail cell; my
{home|prison|purgatory}{! for the past eight months| these days}.
I'm going to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time

```
=== bench ===
I'm {&sitting|lying} on the bench of my jail cell; my
{home|prison|purgatory}{! for the past eight months| these days}.
I'{~m going|ve decided|m trying} to...
// Rest of sitting on bench code goes here
```

fppt.com

Special Text – Variable

- Second, let's add *variable* text
  - Sequences in curly braces { } with bars | show different text each time
  - Tilde ~ sequences show *random* entries

```
=== bench ===
I'm {&sitting|lying} on the bench of my jail cell; my
{home|prison|purgatory}{! for the past eight months| these days}.
I'{~m going|ve decided|m trying} to...
// Rest of sitting on bench code goes here
```

fppt.com

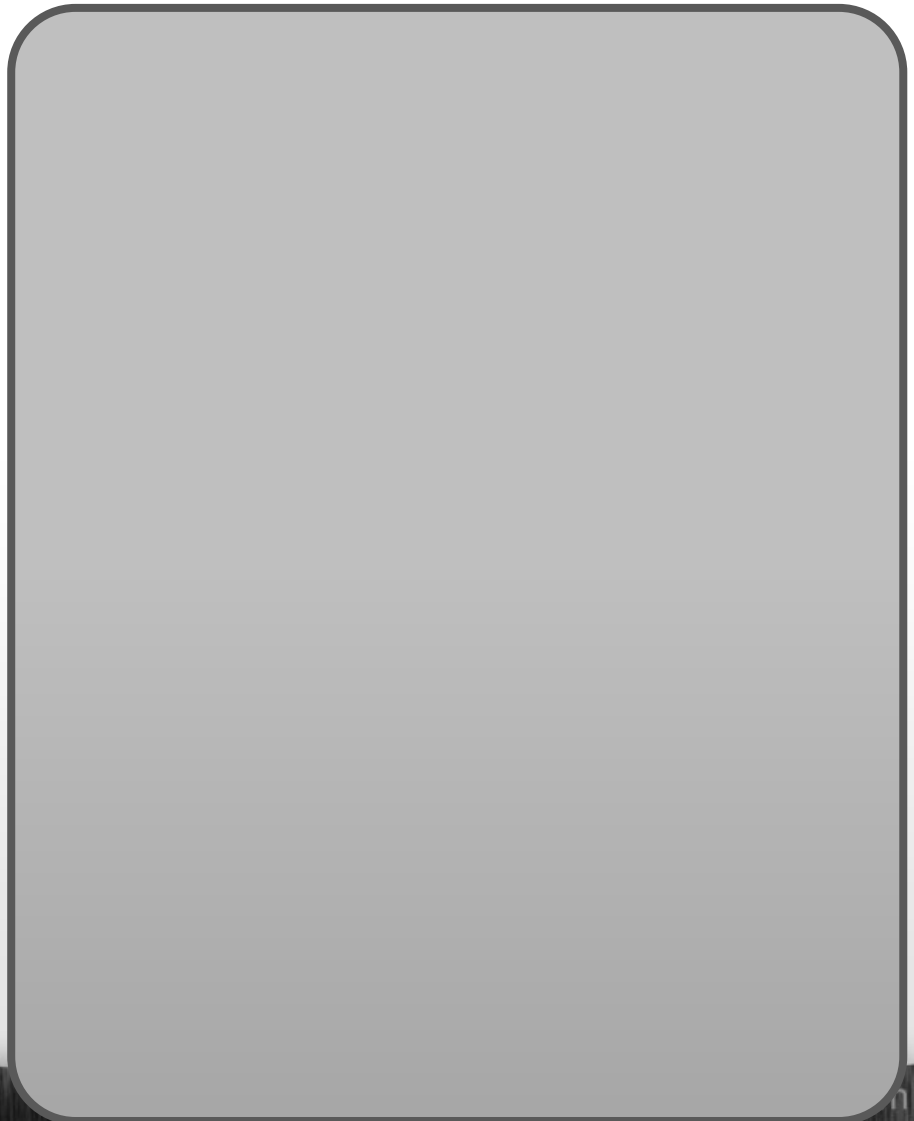
Outline

1. What is *"narrative programming"*?
2. Introducing ***ink***
3. The Basics
4. Better Narrative Flow
5. **Logic & Computation**
6. Conclusion

fppt.com

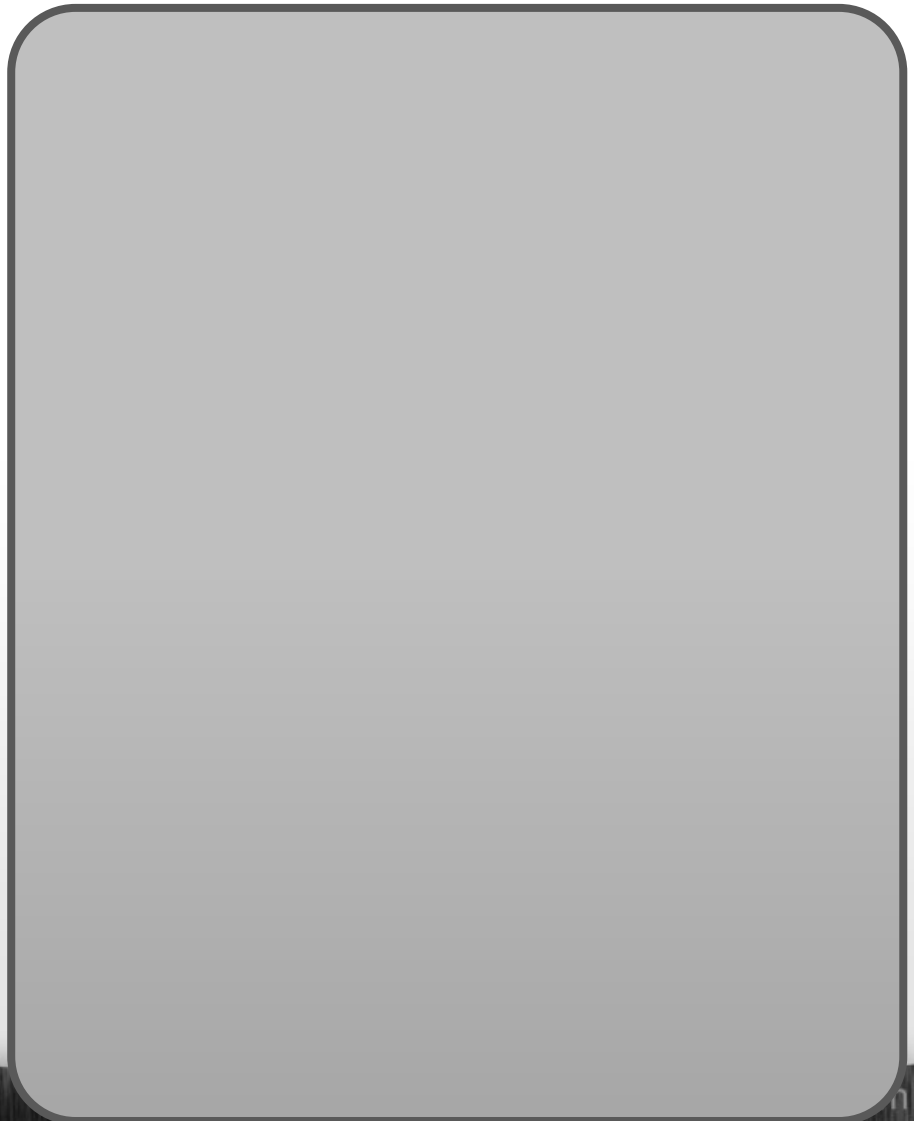
Logic & Computation

- Remember that ***ink*** is a programming language!



Logic & Computation

- Remember that ***ink*** is a programming language!
  - As such, it has a variety of functions for logic and computation



Logic & Computation

- For example, it has *logical* operations
  - e.g. `not`, `&&`, `||`, `==`, `<=`, etc.

```
-> start
=== start ===
You must look both ways
 + Look left.       -> left
 + Look right.      -> right
 * Cross the road. -> cross

=== left ===
-> start
=== right ===
-> start

=== cross ===
{left && right:
  You're across!
- else:
  You get splatted.
} -> END
```

Logic & Computation

- For example, it has *logical* operations
  - e.g. `not`, `&&`, `||`, `==`, `<=`, etc.
- As well as some library *functions*
  - e.g. `TURNS_SINCE(-> knot)`, `TURNS()`, `RANDOM(min,max)` etc.

```
-> start
=== start ===
You must look left then right.
 + Look left.       -> left
 + Look right.      -> right
 * Cross the road. -> cross

=== left ===
-> start
=== right ===
-> start

=== cross ===
{TURNS_SINCE(-> right) == 1 &&
TURNS_SINCE(-> left) == 2:
You're across!
- else:
  You get splatted.
} -> END
```

Logic & Computa

- You can also set , manipulate and print number *variables*…

```
VAR choice = 0
VAR dice   = 0

"Pick a number," the wise man asked.
 * [1]
   ~ choice = 1
 * [2]
   ~ choice = 2
 * [3]
   ~ choice = 3
 * [4]
   ~ choice = 4
 * [5]
   ~ choice = 5
 * [6]
   ~ choice = 6

- The wise man rolls the dice...
 * Wait for it... []It stops.

~ dice = RANDOM(1,6)

A {dice}.

You <>
{dice == choice:
  win!
- else:
  lose.
}
```

Logic & Computa

- You can also set , manipulate and print number *variables*…
- And you can even define your own functions!

```
VAR str = 3
VAR int = 2
VAR cha = -1

How will you get out of this one?
 * [Punch the guard.]
   {Check(str,18):
     -> win
   - else:
     -> lose
   }
 * [Outsmart the guard.]
   {Check(int,10):
     -> win
   - else:
     -> lose
   }
 * [Punch the guard.]
   {Check(cha,15):
     -> win
   - else:
     -> lose
   }

=== function Check(mod,target) ===
~ temp dice = RANDOM(1,20)
You rolled a {dice}, so your score is
{dice + mod}.
~ return dice + mod >= target
```

Logic & Computation

- Thinking about *narrative* flow is not too different to thinking about *program* flow

fppt.com

Logic & Computation

- Thinking about *narrative* flow is not too different to thinking about *program* flow
  - By combining clever narrative with good programming, we can create some cool things in ***ink***

fppt.com

Logic & Computation

- Thinking about *narrative* flow is not too different to thinking about *program* flow
  - By combining clever narrative with good programming, we can create some cool things in ***ink***
- ***So now it's over to you…***

fppt.com

Here are a few ideas to experiment with in ***ink*** for the rest of the session:

1. Try creating a basic escape-the-room game
2. Write an interactive story
3. Write a chat-bot with lots of varied responses
4. Program a logic game (e.g. noughts and crosses or Connect 4)
5. Combine narrative and programming to make something totally unexpected!

Additional help with ***ink***: https://tinyurl.com/inkdocs

fppt.com

Outline

1. What is *"narrative programming"*?
2. Introducing ***ink***
3. The Basics
4. Better Narrative Flow
5. Logic & Computation
6. **Conclusion**

fppt.com

I hope you've enjoyed this session diving into the world of narrative programming!

fppt.com

I hope you've enjoyed this session diving into the world of narrative programming!

We've (hopefully) learned…

fppt.com

I hope you've enjoyed this session diving into the world of narrative programming!

We've (hopefully) learned…

- …that programming can be creative.

fppt.com

I hope you've enjoyed this session diving into the world of narrative programming!

We've (hopefully) learned…

- …that programming can be creative.
- …that the flow of a story and a program are not dissimilar.

fppt.com

I hope you've enjoyed this session diving into the world of narrative programming!

We've (hopefully) learned…

- …that programming can be creative.
- …that the flow of a story and a program are not dissimilar.
- …that experimenting with bespoke languages such as ***ink*** can give us knowledge that we can then apply to other languages.

fppt.com

If you want to know more about ***inkle***:

https://www.inklestudios.com/

Special thanks to Jon Ingold & Joseph Humfrey for designing ***ink***.

Have a great rest of your week!

fppt.com