Branching on Data
Objectives

By the end of today you should be able to...

1. Control program flow with data
2. Understand
3. Scale your games to match the window they are in
Games writing is like:

I write a scene in which a character takes something out of her purse.

Two days later I discover that I caused a shockwave through like six departments.

11:02 AM · Aug 14, 2019 · Twitter Web App

1.2K Retweets 6K Likes

Sarah Arellano (SugarVenom) @TheSugarVenom
Replying to @TheSugarVenom
Project Management: The purse is not on the asset list!
Concept Art: Should it be a basic purse or a couture purse? Is anything else in the purse?
Animation: Does the purse need to move? Do we need a custom animation for her reaching in and removing an item?

Sarah Arellano (SugarVenom) @TheSugarVenom
Design/Props: Does it need to exist in the gameplay environment or only in cinematics?
Narrative: Wouldn’t a woman who carries a purse always have it on her?

Sarah Arellano (SugarVenom) @TheSugarVenom
in short, by mid-development on a project, you must consider with care all verbs, lest you make animators cry, and eliminate all nouns, lest you set back production for two full days.
The door problem

http://www.lizengland.com/blog/2014/04/the-door-problem/

THE DOOR PROBLEM

“THE DOOR PROBLEM”

“So what does a game designer do? Are you an artist? Do you design characters and write the story? Or no, wait, you’re a programmer?”

Game design is one of those nebulous terms to people outside the game industry that’s about as clear as the “astrophysicist” job title is to me. It’s also my job, so I find myself explaining what game design means to a lot of people from different backgrounds, some of whom don’t know anything about games.

The Door Problem

I like to describe my job in terms of “The Door Problem”.

Premise: You are making a game.

- Are there doors in your game?
- Can the player open them?
- Can the player open every door in the game?
- Or are some doors for decoration?
- How does the player know the difference?
- Are doors you can open green and ones you can’t red? Is there trash piled up in front of doors you can’t use? Did you just remove the doorknobs and call it a day?
- Can doors be locked and unlocked?
- What tells a player a door is locked and will open, as opposed to a door that they will never open?
Structuring Your Program
Model View Controller

Model
data and logic

View
representation of the model

Controller
converts input into commands

User
Observer

subject
list of functions
can add and remove from the list
calls functions on the list when thing happens

observer
when this function is called, react however we want

observer
when this function is called, react however we want

https://gameprogrammingpatterns.com/observer.html
Observer - you've seen this with collisions!

A collision has happened!
Notify the callback functions

(\text{player, coin})
Increase the score
Remove the coin

(\text{player, enemy})
\text{decrease the player's hitpoints}

https://gameprogrammingpatterns.com/observer.html
Inherited Classes

The `npc` class inherits from `gameObject`.

Old, monolithic way to design a game engine

- `game object`
  - Physics calculations
  - Input handling
  - Rendering
  - Timing
  - Position

- `npc`
  - Animation
  - Hit Points
  - AI
Entity Component System

- **game object** contains components, each of which handles a small, isolated part.

  - **physics body**
  - **animation**
  - **input handler**
  - **event handler**
  - **drawing**
  - **health system**

A **modular** approach, composing objects out of smaller parts.
Branching Narrative
Problem Solving

Define the actual problem

Think about it

Plan a solution, including alternate plans

Carry out the plan

Look Back: verify you solved the original problem.
Defining the problem

We want our game to have a branching narrative. What do we mean by that?
...most computer games are the combination of two different ways of presenting the player with a challenge, one which I will term emergence (simple rules combining, leading to variation) and one of progression (serially introduced challenges).

Jesper Juul
"The Open and the Closed: Games of Emergence and Games of Progression"
https://www.jesperjuul.net/text/openandtheclosed.html
Structure: triggers embedded in emergent world

- event trigger
- narrative

Unlocks
Structure: progression tree

narrative

narrative

narrative

narrative

narrative

narrative

unlocks
What is the difference?

The Stanley Parable (2013)

howling dogs (2012)

A room of dark metal. Fluorescent lights embedded in the ceiling.

The activity room is in the north wall. The lavatory entrance, west, next to the trash disposal and the nutrient dispensers. The sanity room is in the east wall.

Her photograph is pinned to the side of your bunk. A red LCD reads 367 a few inches over.
What is the difference?

Skyrim (2011)

A room of dark metal. Fluorescent lights embedded in the ceiling.

The activity room is in the north wall. The lavatory entrance, west, next to the trash disposal and the nutrient dispensers. The sanity room is in the east wall.

Her photograph is pinned to the side of your bunk. A red LCD reads 367 a few inches over.

howling dogs (2012)
Structure: events walking progression tree

- Event
- Trigger
- Narrative
- Unlocks
Structure: events walking progression tree
Define the actual problem

Think about it

Plan a solution, including alternate plans

Carry out the plan

Look Back: verify you solved the original problem.
Our Use Case

● What we need out of our data
  ○ Nodes
  ○ Links

● (CS 101: Vertices and Edges)

● One way to organize the data:
  ○ Nodes: Dialogue Objects
  ○ Links: Prereqs

● Alternative:
  ○ Nodes: Agents
  ○ Links: Dialogue Objects between agents
Other Narrative Tools

Yarn: [https://github.com/InfiniteAmmoInc/Yarn](https://github.com/InfiniteAmmoInc/Yarn)

Javascript port of Yarn: [https://github.com/jhayley/bondage.js/](https://github.com/jhayley/bondage.js/)

Ink (from Inkle): [https://www.inklestudios.com/ink/](https://www.inklestudios.com/ink/)

Javascript port of Ink: [https://github.com/y-lohse/inkjs](https://github.com/y-lohse/inkjs)
Problem Solving

Define the actual problem

Think about it

Plan a solution, including alternate plans

Carry out the plan

Look Back: verify you solved the original problem.
https://github.com/ikarth/ink_and_phaser
Scaling

Brief lecture about how to scale your entire game goes here (see other slides).
Problem Solving

Define the actual problem

Think about it

Plan a solution, including alternate plans

Carry out the plan

Look Back: verify you solved the original problem.
More Debugging Tips
Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
   a. assert()
   b. Keep a debugging notebook

2. Make debug tools
   a. Quicker feedback is better
   b. Display values live if possible

3. Only make one change at a time and then test it

4. Just because you paused the game doesn't mean it's paused
   a. And stopping one update doesn't mean you stopped all of them

5. console.log() is slow
   a. Faster to print an array as a string than to individually print the contents
Useful random debugging advice

Walk through your code step by step, explaining to yourself what is supposed to happen
Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
   a. assert()
   b. Keep a debugging notebook

2. Make debug tools
   a. Quicker feedback is better
   b. Display values live if possible

3. Only make one change at a time and then test it

4. Just because you paused the game doesn't mean it's paused
   a. And stopping one update doesn't mean you stopped all of them

5. console.log() is slow
   a. Faster to print an array as a string than to individually print the contents
Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
   a. assert()
   b. Keep a debugging notebook
2. Make debug tools
   a. Quicker feedback is better
   b. Display values live if possible
3. Only make one change at a time and then test it
4. Just because you paused the game doesn't mean it's paused
   a. And stopping one update doesn't mean you stopped all of them
5. console.log() is slow
   a. Faster to print an array as a string than to individually print the contents
Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
   a. assert()
   b. Keep a debugging notebook

2. Make debug tools
   a. Quicker feedback is better
   b. Display values live if possible

3. Only make one change at a time and then test it

4. Just because you paused the game doesn't mean it's paused
   a. And stopping one update doesn't mean you stopped all of them

5. console.log() is slow
   a. Faster to print an array as a string than to individually print the contents
Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
   a. assert()
   b. Keep a debugging notebook

2. Make debug tools
   a. Quicker feedback is better
   b. Display values live if possible

3. Only make one change at a time and then test it

4. Just because you paused the game doesn't mean it's paused
   a. And stopping one update doesn't mean you stopped all of them

5. console.log() is slow
   a. Faster to print an array as a string than to individually print the contents
AABB characters and slopes

An example of a real-world physics-and-debugging problem in a game with 2D physics like yours

https://twitter.com/eevee/status/1133248372624613376