How to Design Worlds

CS 5010 Program Design Paradigms "Bootcamp"

Lesson 3.2

Introduction

- In this lesson, we will show to how to decide whether information should be represented as part of the world state, or whether it should be kept as a constant in the program
  - By the end of this lesson, you should be able to analyze information to determine whether it should be constant or part of the world state.

Review: the 2htdp/universe module

- Provides a way of creating and running an interactive machine.

- Machine will have some *state.*

- Machine can respond to *inputs*.

- Response to input is described as a *function*.

- Machine can show its state as a *scene.*

- We will use this to create interactive animations.

3

What's in the state?

- The state consists of the information that changes in response to a stimulus.

- Other things are constants.

Traffic Light Example

- Number of ticks until next change:
  - this changes on every tick.
  - So: this goes in the state
- Current Color
  - changes when countdown timer runs out
  - So: this goes in the state
- How often each color lasts before changing
  - doesn't change
  - So: this will be a constant, not part of the state

Information Analysis for falling-cat

- dimensions of canvas

- x pos of cat

- y pos of cat

- current speed of cat

- image/size of cat

- acceleration of gravity g

which of these belong in the world?

which should be constants?

which need not be represented?

Information Analysis for falling-cat

- dimensions of canvas
- x pos of cat
- y pos of cat
- current speed of cat
- image/size of cat
- acceleration of gravity g

These vary from frame to frame in the animation, so they are part of the animation's state (the world)

Information Analysis for falling-cat

- **dimensions of canvas**

- **x pos of cat**

- y pos of cat

- current speed of cat

- **image/size of cat**

- *acceleration of gravity g*

These do NOT vary from frame to frame in the animation, so they are NOT part of the animation's state (the world). We will represent them as constants in our program

This is not used in our program, so we don't need to represent it at all

The information analysis depends on the application

- The choice of information to represent depends on what our application does.
- Here are four variations of falling-cat:
  - falling-cat-0: like falling-cat, but doesn't respond to space key
  - falling-cat: our pausable falling cat
  - draggable-cat: like falling-cat, but the cat can be dragged with the mouse
  - model gravity: like falling-cat or draggable-cat, but the cat accelerates with simulated gravity.
- Let's see how the information analysis differs as we change the application.

Info analysis

either **CATSPEED** or 0; represented by **paused?**

	falling-cat-1	falling-cat-2 (pausable)	drag w/ mouse	model gravity
dimensions of canvas	constant	constant	constant	constant
x pos of cat	constant	constant	world	world if draggable
y pos of cat	world	world	world	world
current speed of cat	constant	world	world	world
image/size of cat	constant	constant	constant	constant
acceleration	not represented	not represented	not represented	constant
Fields needed:	y-pos	y-pos, paused?	x-pos, y-pos, paused?	x-pos, y-pos, current-speed, paused?

Gravity and Speed

- When there is no gravity, the cat's speed is either 0 or CATSPEED, so we can represent this with one bit (paused?).

- When there is gravity, the cat accelerates, so we need to keep track of its current speed.

- When the cat resumes motion, it should resume its previous speed.

- So we need to keep track of both its paused? state and its previous velocity.

The problem should specify what happens when the cat resumes falling. Here we've assumed that the cat resumes falling at the same speed it was moving when it paused. If we specified that the cat starts falling again with speed 0, how would that change the representation?

Recipe for Designing a World

Here are the steps in designing a world. These are the same steps we followed in falling-cat.

How to Design Universe Programs

1. Information Analysis
   • What events should the world respond to?
   • What information changes in response to an event?
   • What information doesn't change in response to an event?

2. From your information analysis, write out the constant definitions and data definitions.
3. From your list of events, write a wishlist of functions to be designed
4. Design the functions on your wishlist (use the design recipe!)

Summary

- Information that can change after an event goes into the world state

- Info that doesn't change is represented by constants

- Manage your project with a wishlist

- Wishlist must include contracts and purpose statements for each function

Next steps

- If you have questions about this lesson, post them on the Discussion Board

- Do the Guided Practice for this Lesson

- Go on to the next lesson.