Functions for Character Animation

(Also available in WeScheme)

Students define functions that control the movement of the target and danger in their games.

Lesson Goals	Students will be able to: • Apply the ***Design Recipe*** to create a ***function*** given the constraints of a word problem. • Explain the basics of animation.
Student-Facing Lesson Goals	• Let's learn about animation. • Let's use the Design Recipe to help us breakdown word problems that will get our game characters moving!
Prerequisites	• Simple Data Types • Contracts • Functions: Contracts, Examples & Definitions
Materials	• PDF of all Handouts and Page • Blank Game Starter File • Lesson Slides • Printable Lesson Plan (a PDF of this web page)
Key Points for the Facilitator	• Encourage students to take their time in understanding *why* we want to fix `update-danger` and `update-target`. • Students might be confused as to *how* the animation is working. The `make-game` function at the bottom of the file has many inputs - including `update-danger` and `update-target`. `make-game` takes in all those inputs, including the functions we'll write, and creates the interactive window that we see when we click the "Run" button!

Glossary

coordinate :: a number describing an object's location

design recipe :: a sequence of steps that helps people document, test, and write functions

domain :: the type or set of inputs a function expects, i.e., the independent variable(s) that govern the output of the function

function :: a relation from a set of inputs to a set of possible outputs, where each input is related to exactly one output

range :: the type or set of outputs that a function produces, i.e., the dependent variable(s)

Animation *45 minutes*

Overview

Students connect the behavior of functions with changing coordinate values, ultimately leading to animation.

Launch

- How does a flip-book animation work?
  - *Each page of the book is slightly different, and the pages go so fast that the motion looks smooth.*
- Why do we see movement from still images?
  - *Our eyes fill in the gaps between rapidly changing images.*
- How might this apply to our game?
  - *If we change image* ***coordinates*** *a little bit at a time, they will appear to move.*

- Draw a number line on the board, running from 0 to 1000 (you can also lay tape on the floor, or use a tile floor as a coordinate plane!).
- We are going to write a function, called `update-target`, that will be responsible for changing the x-coordinate of the `TARGET` location.
  Note: In our game, there is hidden code that will randomly generate the y-coordinate for where the `TARGET` *enters the screen. Our* `update-target` *function will take in that coordinate, but won't change it. For this role play, it will likely make the most sense to work with a y-coordinate of 0.*
- Before we write the function, let's act it out!
- Select a student volunteer to be `TARGET` and pick a starting location for our `TARGET`, eg. (300, 0) .
- For this roleplay, we will have our `update-target` function move our `TARGET` left across the x-axis by 50 (pixels) each time the function is called.
  - When the `TARGET` hears "update target" followed by their current location, they take a step in the negative direction, moving left across the x-axis by 50 (pixels).
  - Make `TARGET` move by calling out the function and starting location, eg. `update-target(300, 0)`.

- Make `TARGET` move several more times by calling out the function and new location, eg. `update-target(250, 0)`, etc.

- Ask students: How could we change our `update-target` function to move the `TARGET` across the classroom *more quickly?*
  - *We could have the* `TARGET` *move by more pixels each time, for example by 100 instead of 50.*
- What did you notice about the movement of TARGET? What was changing?
  - *Answers will vary: they were moving horizontally, their x-coordinates were changing, they were moving one step at a time, etc.*
- If we want to write another function called `update-danger` and have it move the `DANGER` 50 pixels to the right, how will the function work?
  - *It will take in the x and y-coordinate and add 50 to x.*

Investigate

- Sign in to code.pyret.org (CPO) and open your saved Game Starter Files, or make a new copy.
- Examine the `update-danger` function. Identify the Contract, and interpret what the function is currently doing.
- Complete the first word problem on Danger and Target Movement.
- Transfer the code to your Game Starter File.

When students click the "Run" button, the working `update-danger` function should automatically move the `DANGER` image across the screen!

- Complete the second word problem on Danger and Target Movement.
- With your partner, transfer the code to your Game Starter File.
- Click "Run" to see `DANGER` and `TARGET` move across the screen independently!

Extension Activities

Once students have successfully gotten `update-target` and `update-danger` working, they can change the functions to make the characters move in whichever direction and whatever speed they want to make the game the most fun to play! They should be sure to modify their purpose statements and examples if they change their functions.

Synthesize

Connecting the code to the underlying math is important - especially if you want to customize your game!

- What part of the function controls the character's *speed?*
- What part of the function controls the character's *direction?*
- If you wanted the characters to move in 2 dimensions (diagonally, for example), would anything have to change about the ***Domain***? What about the ***Range***?