

Lesson 2 - Logical Thinking - Teacher's Guide

<http://www.stencyl.com/teach/act2/>

Objective

Introduce students to “logical thinking” through programming. Students will learn the following concepts:

- How do computers work?
- Control Flow (how a program runs)
- Conditionals (if, else, else-if, boolean statements)
- Logical Operators (equals, and, or, not)

Outcome

Students will create a 4-direction movement script from scratch. In this process, students will translate real-world requirements into instructions.

Lesson Plan (1 hour)

Discussion
15 minutes

Cover the topics under Discussion Notes (Page 2)

Present the topics. Pose questions at appropriate points and encourage students to participate in the discussion.

Demo
20 minutes

Demonstrate the Logic Designer to the class (Pages 3 - end)

Follow our template.

Alternate Plan: *You may prefer to alternate between demoing and having students complete parts of the activity.*

Activity
25 minutes

Make a Game From Scratch

Students will build a game entirely from scratch using the concepts they've just learned.

Note: If students don't finish, let them finish it as homework or during the next class. Either way, we recommend allowing an extra class to have students complete some or all of the extra activities.

Discussion Notes

Topic 1: What can computers do?

Computers power everything from the obvious (PCs, tablets, smartphones) to the not so obvious (cars, planes, medical devices).

Discussion: Have students come up with examples and have them explain what role the computer plays in powering those examples.

Topic 2: How do you tell a computer what to do?

We make *programs*. Programs are like recipes. They are instructions that tell a computer exactly what to do.

Discussion: Wouldn't that mean that a program does the same thing each time it's run? What can you do to change the outcome? (User input from keyboard/mouse/controller)

Topic 3: Conditionals

Conditionals let us change the course of a program. They are like asking a yes-no question.

Example: If the hero falls off the bottom of the stage, play the death music and restart the level.

Conditionals come in several forms:

- if
- if-else
- if-else/if-else

Topic 4: Logical Operators

Every conditional contains *statements*. Statements always come out to yes or no for an answer. Logical operators let us combine several statements into a bigger one.

- and
- or
- not

Discussion: Analyze examples from real games to make this all concrete.

How to Demo the Logic Designer (20 minutes) - (ROUGH DRAFT - NEEDS EDIT)

Demo steps are divided into a description and accompanying actions. The **descriptions** have you (the teacher) **explain** what's happening. The **actions** are **what you do on the computer** to demonstrate how to use Stencyl.

Part 1 of 4: Introduction

Description

Stencyl's Logic Designer (also referred to as Design Mode) lets users write code without having to worry about the syntax of a formal programming language. Instead of typing code, users drag and drop colorful bricks together.



Actions

None. Just describe what our Logic Designer is and does.

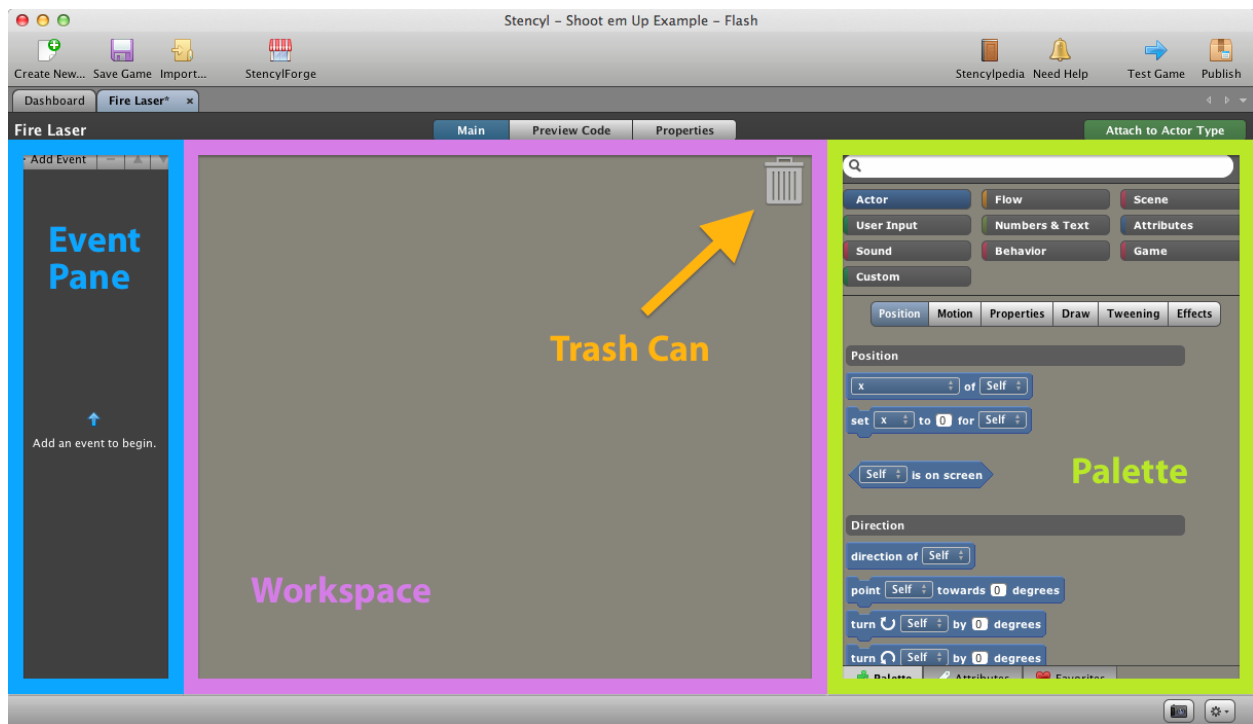
Part 2 of 4: Basic Operation

Explanation

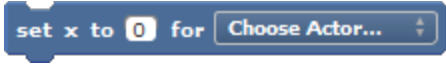



In this section, you'll run through the major parts of the Logic Designer and describe the basics of using it.

Explanation (Part 1) - The Interface

Event Pane	A list of all of a Behavior's events. Each event is associated with its own separate workspace.
Workspace	Where you drag in the blocks that form the logic for an event. Also contains a Trash Can for disposing of unwanted blocks.
Palette	The area on the right where you pick what blocks to add to the Workspace. Split up into Categories (the buttons at the top) and Pages within a Category. Can use the Search field to filter blocks by name.



Explanation (Part 2) - How to Use

Adding Blocks	Drag and drop blocks from the Palette (on the right) into the Workspace.
Moving Blocks	Click and drag blocks around in the Workspace to move them.
Moving a Single Block	Holding down SHIFT while dragging will let you pull out just the selected block, versus the default behavior, which is to pull out the block and every block below it.
Removing Blocks	Drag blocks to the Trash Can to remove them. (Alternatively, right-click and select Remove.)
Copying Blocks	Holding down ALT while dragging will duplicate the currently selected block. If you wish to copy a group of blocks, drag the desired group out, right-click and select Duplicate.
Types of Blocks	<p>Action Blocks have jigsaw puzzle notches on top and bottom.</p>  <p>Normal Blocks have no notches. They are placed into the fields (and sometimes, dropdowns) of other blocks.</p>  <p>Some normal blocks look like hexagons.</p>  <p>These denote Boolean (Yes/No) blocks that fit, predictably, into fields that also look like hexagons.</p> 

Actions

None. Instead, run through the information above and demonstrate as you go along.

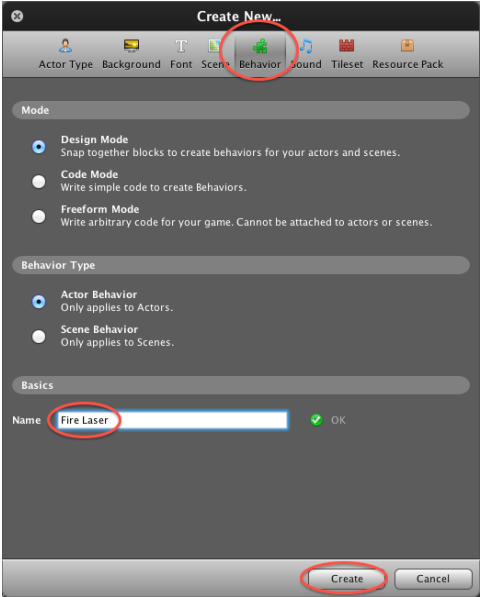
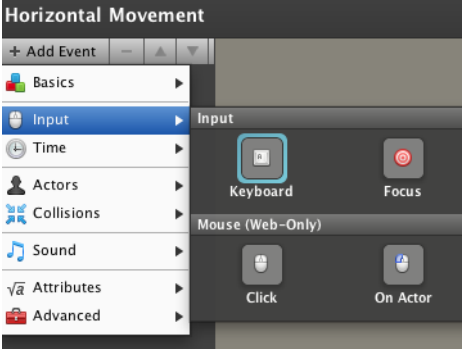
Part 3 of 4: Events

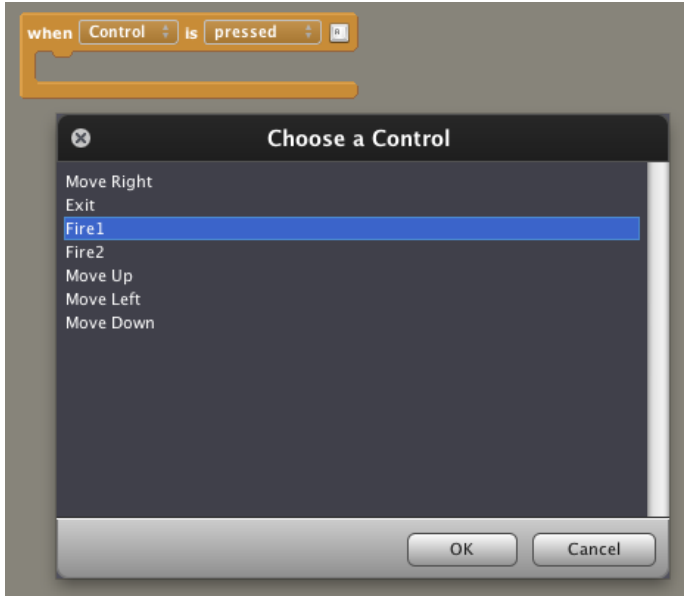
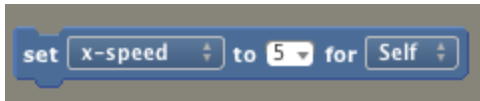
Explanation

Events are things that happen in your game that trigger some kind of action, or response. Examples of actions include keyboard presses, mouse clicks and collisions.

You'll now demo Events to students using Project 2.

Actions

1	Open up Project 2 .
2	<p>Create a new Behavior. Name it anything you'd like.</p> 
3	<p>Click on Add Event. Talk about the different kinds of Events that are available.</p> <p>Be sure to mention that Basics > When Updating is the most important event. It is the “event” that “always” happens, so it is the default way to run logic in a game.</p> 

	Once you're done explaining, pick out the Input > Keyboard event.
4	<p>Explain how Keyboard events are fired when a key is pressed or released.</p>  <p>Keyboard events are mapped to controls. Instead of mapping directly to specific keys on a keyboard, we work with controls instead.</p> <p>Controls let you map events to abstract, human-friendly actions. If you ever decide to change the key, you just need to change the control, rather than every event. Isn't that convenient?</p>
5	Inside the Keyboard Event, click on Control > Choose Control . Then, pick out the Fire1 control.
6	<p>Inside the body of the event, insert the following.</p> 
7	Finally, attach the Event to the Hero actor.
8	<p>Run the game. Press the Spacebar (the key that the Fire1 control is mapped to).</p> <p>As you'd expect, the Hero will move forward (without stopping).</p>
9	Optional - Add a second Keyboard Event that will stop the Hero when you release the key.

Optional: If you feel it would be useful to run through the creation of a simple behavior from start to finish, this would be a good time to do that before students begin the activity.

Part 4 of 4: Begin the Activity

Take questions and turn students over the activity.

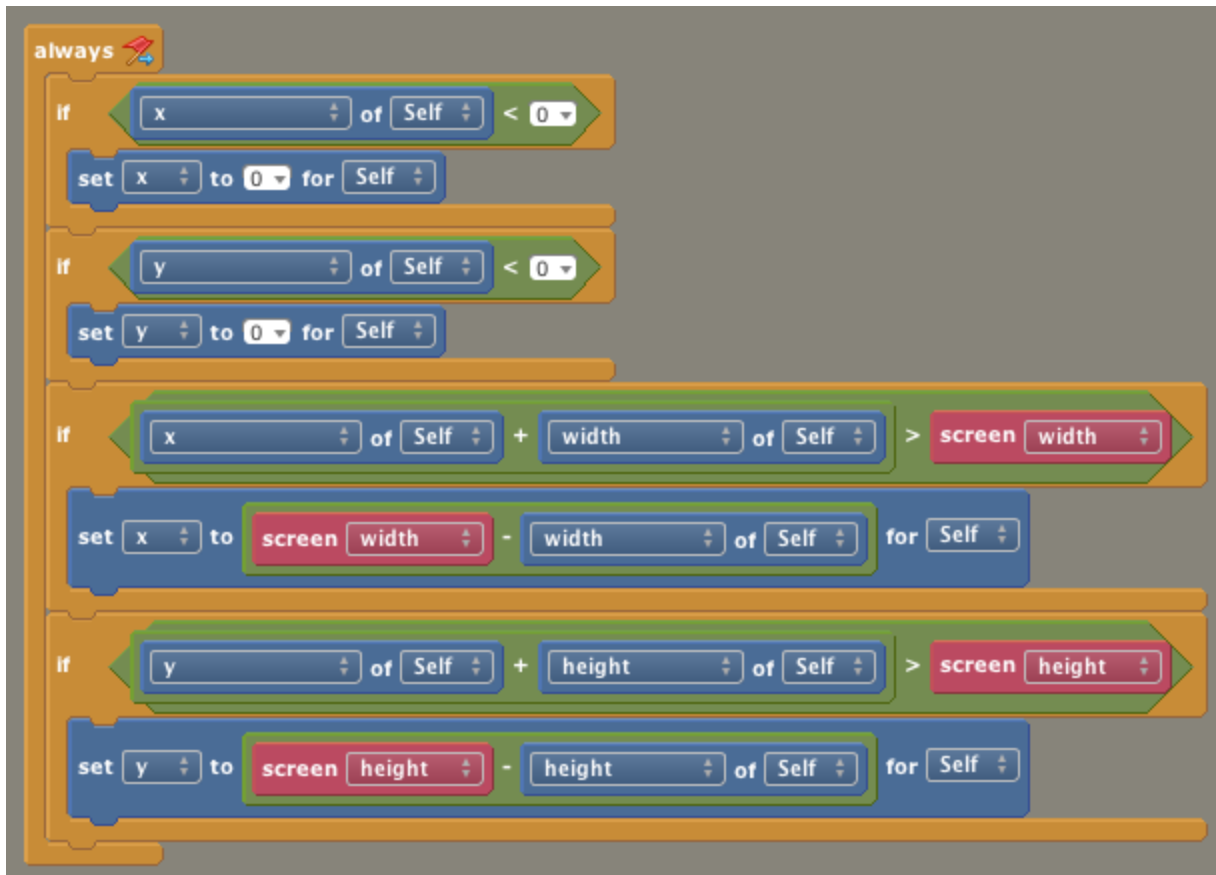
IGNORE EVERYTHING BELOW THIS POINT

Solution to the Main Activity



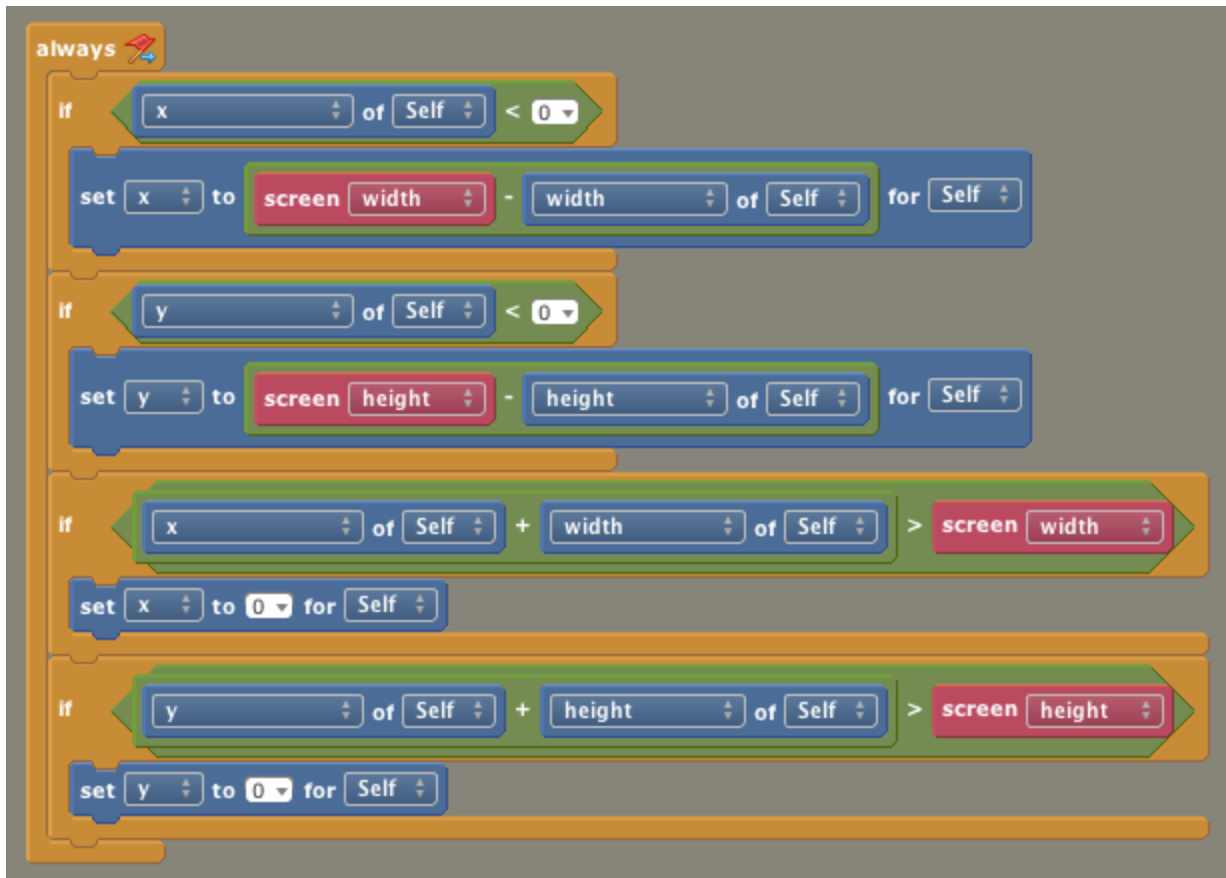
TODO: Explanation?

Solution to "I'm Trapped"



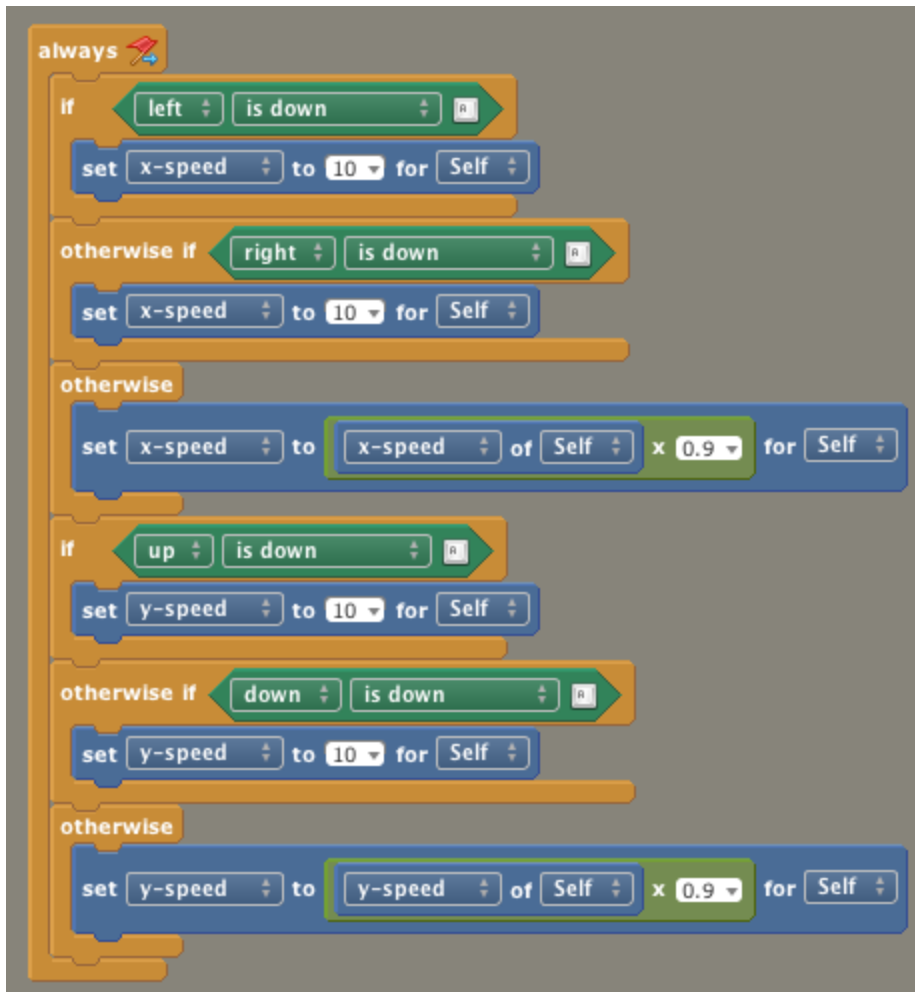
TODO: Explanation?

Solution to "Wrap Around"



TODO: Explanation?

Solution to "Ice"



TODO: Explanation?

Random Thoughts

TODO: Special signup form for educators making common logins, special education alternate "download" that has the edu flag flipped to on by default (maybe even pre-configured to the common login?), track this internally so we know what's going on. Pitch as a summer camp in a box and maybe pilot this in a few schools.

TODO: This kit assumes we can provide a reduced palette in "education" mode.

TODO: academy's mobile app can initially focus on pre-packaged projects vs. trying to be the entire experience.

TODO

- Make the corresponding page for this. Page is web-form of this activity with download link of PDF (for students), download link for the project file, has images that the printable portion lacks. Teacher comments via link to a forum topic.