Independently program a skater game.

For the student who has completed Frogger, this activity tasks him/her with creating a pre-designed game requiring basic programming skills. This can be used as an assessment or as an activity to strengthen independent programming skills with support.

Created by: Susan Miller, University of Colorado, School of Education

This curricula has been designed as part of the Scalable Games Design project.
It was created using portions of prior work completed by Karine Laidly.

This material is based upon work supported by the National Science Foundation under Grant No. DRL-1312129 and CNS-1138526. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Skater Game

Lesson Objective:

For the student who has completed Frogger, this assessment tasks him/her with creating a pre-designed game requiring basic programming skills.

Prerequisite Skills:

- **Students are presumed to have successfully completed a Frogger Game**

Computational Thinking Patterns:

- **Cursor Control**
- **Generate**
- **Absorb**
- **Collision**

Length of Activity:

- **One 30-45 minute lesson**

Activity Description:

- **Assessment – Teacher instructions**
- **Assessment – Student instructions**
- **Assessment – Answer Key**

Table of Contents

Teacher Instructions:
Scalable Game Design Assessment using AgentSheets – Independently program a game:

When used as an assessment tool:

This activity can be used at the end of Frogger to assess the knowledge level of a student.

A. Give each student a set of the Student Instructions.
B. Allow students to talk with the person next to them about the assignment for the first five minutes. This gives students a chance to think through their plan.
C. Students should not work with anyone else after the first five minutes.
   a. You might consider allowing students access to their notes or to prior games they made to use as examples. This would be comparable to a real-life situation where programmers look back at their prior work when creating new games.
D. A possible solution is provided in the answer key. However, they best test is playing the game itself to see if it works as expected. Remember that there are many ways to program the same actions, and students should not be penalized for using different ways that provide the same end result.

When used as an activity:

This activity can be used at the end of Frogger to strengthen a student's ability to program independently, yet still provide support if needed.

A. Give each student a set of the Student Instructions.
B. Allow students to talk with the person next to them about the assignment for the first five minutes. This gives students a chance to think through their plan.
C. Have students work independently for a set amount of time (perhaps 15 minutes). Then allow them to 'phone a friend' (talk with another student in the classroom) if they are stuck. After 5 minutes, have students work independently again.
D. A possible solution is provided in the answer key. However, they best test is playing the game itself to see if it works as expected. Remember that there are many ways to program the same actions, and students should not be penalized for using different ways that provide the same end result.

Student Instructions:
Scalable Game Design Assessment using AgentSheets – Independently program a game:

A. Using AgentSheets, create a new project and select 32x32 pixels for the agent size.
B. Save the project in a folder designated by your teacher.
C. Create the following agents:
   - A person (don't spend too much time on this; 3 minutes is enough time!) Mask the color "white" so it is see-through.
   - Also create a duplicate depiction that is rotated 90 degrees. Call it "FallenPerson".
   - A garage (again, 3 minutes is enough time – this is not art class!) Mask the color "white" so it is see-through.
   - A skateboard (again, 3 minutes!) Mask the color "white" so it is see-through.
   - A wall (this is simply a black square that we will use as a border for the worksheet)

D. Create a new worksheet, save it and call it "skater worksheet".
   - Keep it small, no need to resize the worksheet.
   - With the wall agent, create a whole border for the worksheet (all 4 sides of the worksheet).
   - Add the person in the bottom of the worksheet in the white space (just above the wall).
   - Add the garage on the middle left side of the worksheet by the left wall.
   - Save your worksheet!

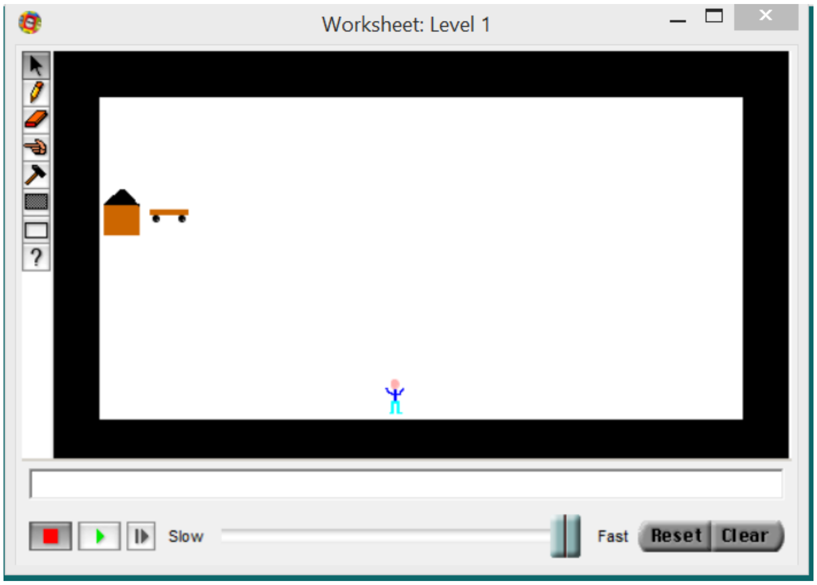
E. Create the following behaviors:
   This game will have the garage generate skateboards that the person can hop onto. If the person runs into the wall while on the skateboard, the game is over. Read below for the details of the behavior.

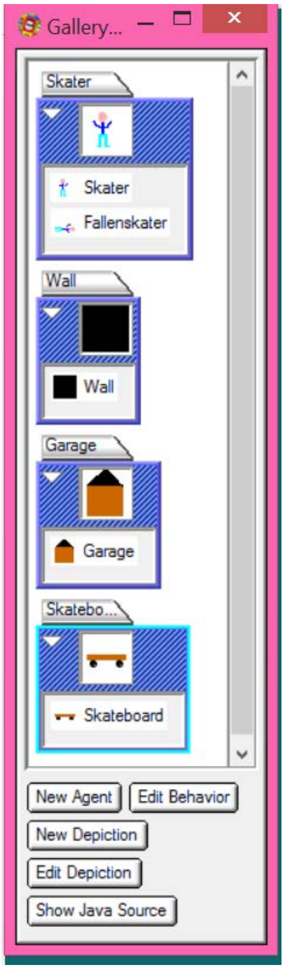
   - Make the garage generate skateboards every 1 second.
   - Make the skateboard move to the right across the worksheet every 0.5 seconds.
   - Make the wall absorb the skateboards once they collide.
   - Make the person be able to hop in all 4 directions using the arrow keys.
   - If the person hops and lands on a skateboard, then he should be able to ride on it as it moves across the worksheet.
   - If the person collides with the wall while riding the skateboard, make him appear like he fell (use the FallenPerson depiction), show a message that says "CRASH!" then reset simulation.

F. Extra Credit! Can you make the game NOT allow the person to go on the wall, so he is stuck inside the wall frame?

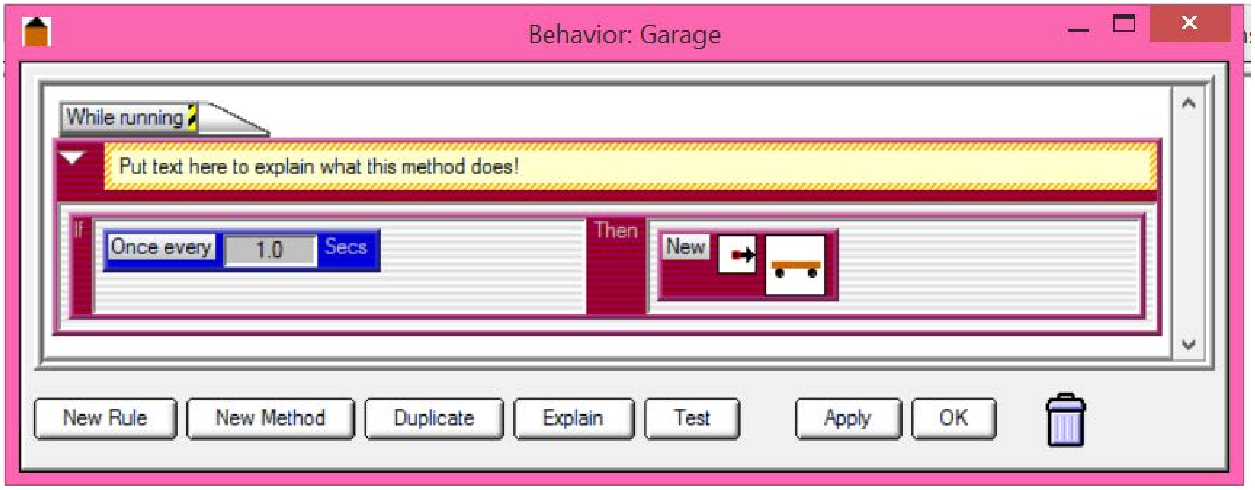
Teacher instructions: Possible Solution

Worksheet and Agents:





Garage Behavior

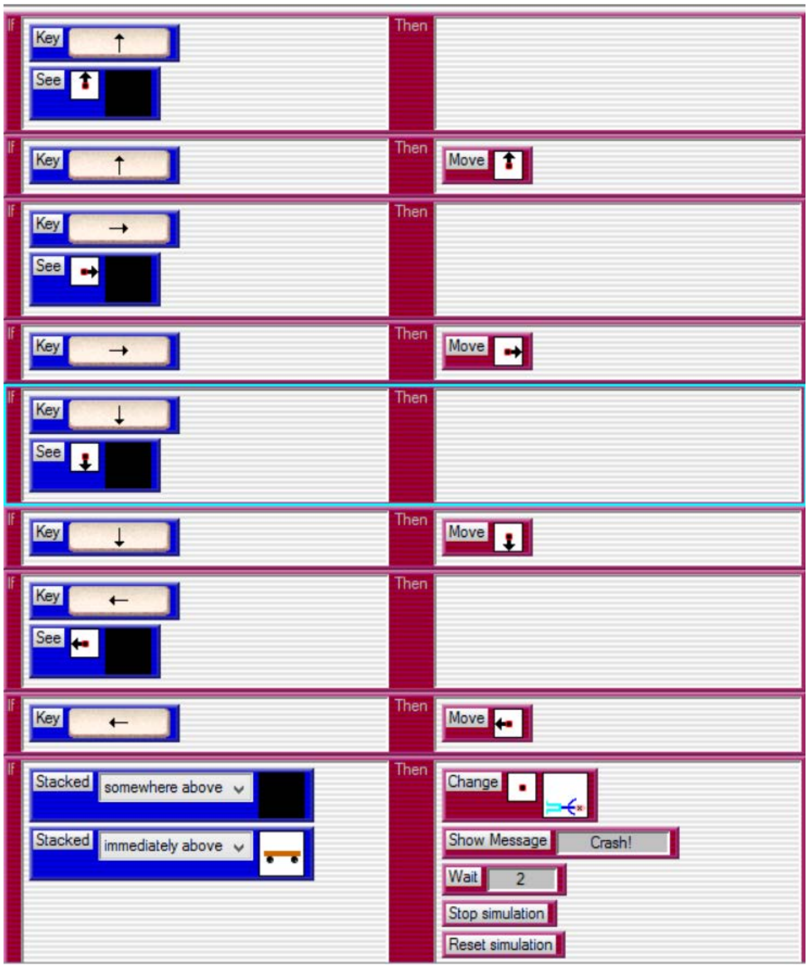


While running

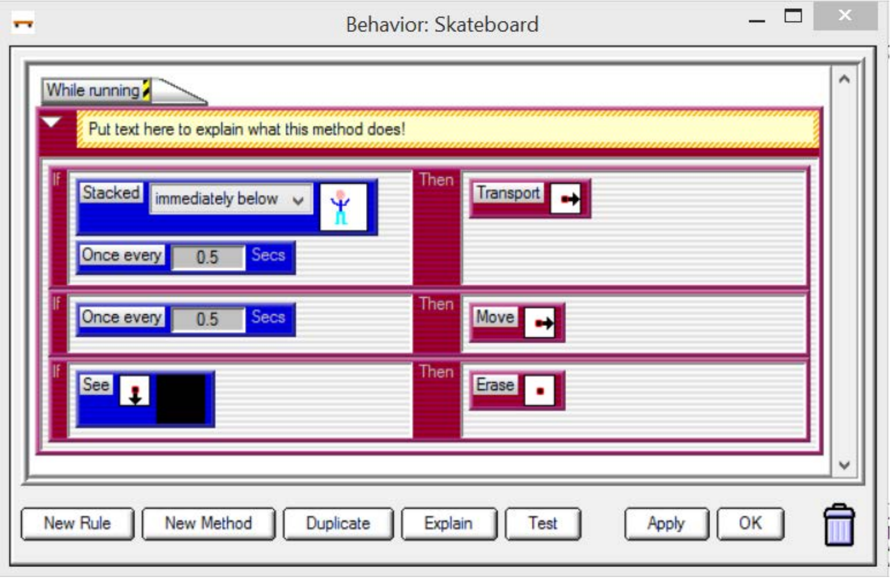
Put text here to explain what this method does!

If Once every 1.0 Secs Then New

Person Behavior



Skateboard Behavior


Behavior: Skateboard
While running
Put text here to explain what this method does!
If
Stacked immediately below
Once every 0.5 Secs
Then
Transport
If
Once every 0.5 Secs
Then
Move
If
See
Then
Erase
New Rule
New Method
Duplicate
Explain
Test
Apply
OK