



# Creating “Frogger”



You are a frog. Your task is simple: hop across a busy highway, dodging cars and trucks, until you get to the edge of a river, where you must keep yourself from drowning by crossing safely to your grotto at the top of the screen by leaping across the backs of turtles and logs.

**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 ideas from and portions of prior work completed by Fred Gluck, Cathy Brand, Carla Hester-Croff, Jason Reub, Sandy Wilder and Mark Shouldice.

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.

## Lesson Objective:

- To create a game of Frogger
- To master the basics in the use of AgentSheet software
- To be able to describe and apply the Computational Thinking Patterns identified below.

## Prerequisite Skills:

- Basics of computer handling
- Identifies hardware components eg. keyboard, mouse, monitor/screen
- Identifies cursor
- Recognizes the typical features of an applications window title bar, toolbar, menu bar, status bar, scroll bar
- Has a knowledge of drop-down menus
- Selects menu items from a drop-down menu
- Starts an application and creates a document
- Names and saves a document in appropriate folder/directory

## Computational Thinking Patterns:

- Cursor Control
- Collision
- Generate
- Absorb
- Transport

## Length of Activity:

- Five 30-45 minute lessons, although some students may advance more quickly

## Activity Description:

- Part 1: Create basic world (worksheet) and the agents
- Part 2: Program the frog, trucks and the tunnel
- Part 3: Create and program the river, logs, and the turtles

Challenge activities for students who finish early

## Table of Contents

<b>Teacher Instructions:</b>	<b>Part 1 – Create Worksheet and Agents</b>
<b>Student Handout 1A:</b>	<b>Part 1a - Create Worksheet</b>
<b>Student Handout 1B:</b>	<b>Part 1b - Create Agents</b>
<b>Student Handout 1C:</b>	<b>Agent Creation Models</b>
<b>Teacher Instructions:</b>	<b>Part 2 – Create Worksheet for the Game</b>
<b>Student Handout 2:</b>	<b>Part 2 – Create Worksheet for the Game</b>
<b>Teacher Instructions:</b>	<b>Part 3a – Understand Agents Behaviors</b>
<b>Student Handout 3:</b>	<b>Part 3a – Understand Agents Behaviors</b>
<b>Teacher Instructions:</b>	<b>Part 4 – Frog Behavior</b>
<b>Teacher Instructions:</b>	<b>Part 4 – Truck Behavior</b>
<b>Teacher Instructions:</b>	<b>Part 4 – Truck/Frog Collision</b>
<b>Student Handout 4:</b>	<b>Agent Behaviors</b>
<b>Teacher Instructions:</b>	<b>Part 5: The River</b>
<b>Student Handout 5:</b>	<b>The River</b>
<b>End of Unit Review Sheet - Frogger</b>	
<b>Student Challenge 1.0</b>	<b>Alligators</b>
<b>Student Challenge 2.0</b>	<b>Prevent Cheating</b>
<b>Student Challenge 3.0</b>	<b>Dodge Cars and Trucks</b>

## Vocabulary/Definitions

*(Found on page 2 in the student packet)*

**Absorb.....**This is the opposite pattern of Generate. Instead of an agent generating other agents, an agent absorbs a flow of other agents in the absorption pattern (i.e. a tunnel absorbing cars), making them ‘disappear’

**Action.....**the requested behavior of an agent if the conditions are true

**Agent.....**a character in the game

**Array.....**a rectangular arrangement of agents

**Collision.....**the situation when two agents physically collide.

**Condition.....**the situation that must be ‘true’ for an action to occur

**Depiction.....**a second image of the original agent. For example, the frog can have two depictions: what it usually looks like, and what it looks like after it has been squished

**Generate.....**the ability to create a new agent. To satisfy this pattern, an agent is required to generate a flow of other agents; for example, cars appearing from a tunnel

**Grotto.....**the land where the flag is located, which must be reached to win the game

**Transport.....**represents the situation when one agent carries another agent; the ability of an agent to be on top of, and move with, another agent

## General Teaching Strategies<sup>1</sup>

### Basic Philosophy

- The educational goal of these lessons is to learn and apply Computational Thinking Patterns in the context of a familiar game. Emphasis on these Computational Thinking Patterns is essential for student understanding.
- Every effort has been made to create instructions with an eye toward guided discovery. Direct instruction has been used for those aspects where students are learning the code for the first time; however, materials have been provided to ensure that students are understanding the programming concepts, as opposed to simply copying code. Note that special materials have been designed for students who are new to this program.
- Student materials are available for each portion of the game design. These materials are intended to be used in addition to teacher materials, which provide prompts and discussion points. Students may become frustrated with too little teacher support. Students may lose out on conceptual understanding with too much teacher support.

### Guided Discovery Process

- **Model the process** rather than just giving students the answer. Building the game on your own, before trying it with your students will enable you to see possible struggling points.
- Have students work through problems on their own. Ask directing questions or give helpful suggestions, but **provide only minimal assistance** and only when needed to overcome obstacles.
- Don't fear **group work!** It is common for computer programmers to talk through problems with one another, and to use code snippets found from other programs, and other programmers. Talking through coding problems enables students to think more

---

<sup>1</sup> This information is supported by research found in the following documents:

Basawapatna, A. R., Koh, K. H., & Repenning, A. (2010, June). Using scalable game design to teach computer science from middle school to graduate school. In Proceedings of the fifteenth annual conference on Innovation and technology in computer science education (pp. 224-228). ACM.

National Research Council. (2011). *Learning science through computer games and simulations*. (M. Hilton & M. Honey, Eds.). Washington, DC: The National Academies Press.

National Research Council. (2014). *STEM Integration in K-12 Education:: Status, Prospects, and an Agenda for Research*. (M. Honey, G. Pearson, & H. Schweingruber, Eds.). Washington, DC: The National Academies Press.

Repenning, A., & Ioannidou, A. (2008, March). Broadening participation through scalable game design. In *ACM SIGCSE Bulletin* (Vol. 40, No. 1, pp. 305-309). ACM.

critically about Computational Thinking Patterns, as well as the steps needed to solve a problem. Additionally, seeing how others solved an issue with code helps students realize that problems often have multiple solution strategies, and some that might be more effective than others

- Recognize that programming is largely a process of **trial and error**, particularly when first learning. It is helpful to encourage this mindset with your students.

## Building Blocks

- Each project is designed to build on the prior one. Very little student support is provided where expertise has already been created. Conversely, material that is new has more support.
- Be sure to talk through the building blocks (especially for PacMan in the area of diffusion and hill climbing) as these Computational Thinking Patterns will appear often in future games and simulations.
- Remember that conceptual understanding takes time, and it may be necessary to explain these concepts multiple times, using different examples, so that all students can be successful.

## Support Learning

- Research shows that game design is associated with engaged students, and engaged students show higher levels on conceptual understanding. Allowing students to personalize their games aids in this engagement and motivation.
- Coding may be difficult for some students, and all students are likely to be frustrated at times when the code does not produce the expected results. **Praise students** for sticking with the troubleshooting process and encourage them to share what they learned with others.
- Be sure to communicate that **the process is more important than the answer**, and that coding of a project often takes time. Do not place pressure on your students to ‘hurry up’ and resort to giving them the code. The process of figuring it out on his/her own will result in much stronger conceptual understanding.

## Differentiated Instruction



*Note that there are many vocabulary words in this lesson that may be new for your students. Take time to define those words. Using the words in context often will reinforce their meaning for the students.*

- **Students who need a challenge:** Some students with more fluency in programming may finish this very quickly – be prepared for them to move on earlier than other students by having student materials ready in advance.
- **Students who need more assistance:** Other students (especially those with no Frogger experience) may struggle a bit more. There are two options for differentiated instruction. Consider the needs of the student and the class as you decide which will work best.
  - Option 1: Pair a struggling student with an experienced student
  - Option 2: Provide student with a tutorial found on the Scalable Game Design Wiki<sup>2</sup>. Note that tutorials do not support independent thinking and should only be used when absolutely needed.
  - Vocabulary for ELL Students: : Generate, Absorb, Collision, Agent, Grotto, Depiction, Condition, Transport
  - Time management issues: While students can be more engaged when they design their own agents, some students can spend too much time on this design or find it frustrating. Handout 1C provides block images of each agent as portrayed in this lesson.

---

<sup>2</sup> [http://sgd.cs.colorado.edu/wiki/Scalable\\_Game\\_Design\\_wiki](http://sgd.cs.colorado.edu/wiki/Scalable_Game_Design_wiki)

## Teacher Instructions:

### Part 1 – Create Worksheet and Agents

Introduce this project to the students by asking them if they know the game, Frogger.

- Ask students to explain how the game works, and the rules of the game.
- You may choose to show this video of the original Atari Frogger
  - <https://www.youtube.com/watch?v=okm0VtF2gH8>

If you show this video, ask your students...

- What does the frog have to do?
- How do you win the game?
- What special features are part of the game?
- Why are some logs blinking?
- What happens when the frog goes in the water?
- What happens when the frog does not cross the water?
- What do you think the alligator does?

Explain that these are all design features that must be considered when planning a game. Now, tell them that they will be designing their own Frogger game.

As a class, briefly create a description of the Frogger game similar to the one below.

- identify game objects, called agents, by locating nouns in the game description

*You are a frog. Your task is simple: hop across a busy highway, dodging cars and trucks who come out of (and go back into) tunnels, until you get to the edge of a river, where you must keep yourself from drowning by crossing safely to your grotto at the top of the screen by leaping across the backs of turtles and logs.*

- categorize agents into user controlled agents (hint the game is called Frogger), agents that move or do other things by themselves (sometimes also called artificial intelligence agents) and completely passive agents acting as props such as the street.

*User controlled agents:*

- *Frog*

*Artificial Intelligence Agents:*

- *highway, cars, trucks, tunnels, river, turtles, and logs, snakes and alligators*



- identify agent interaction by locating verbs in the game description

*You are a frog. Your task is simple: hop across a busy highway, dodging trucks who come out of (and go back into) tunnels, until you finish crossing the river, where you must keep yourself from drowning by crossing safely to your grotto at the top of the screen by leaping across the backs of turtles and logs.*

- Trucks drive along the highway
- Trucks come out from the tunnels and go back into the tunnels
- Logs and turtles float along the river
- Snakes and alligators eat the frog

**This lesson will be teacher-directed in the beginning so that students learn AgentSheets while learning how to program. Be sure to give students time to work on their own and figure things out using the program.**

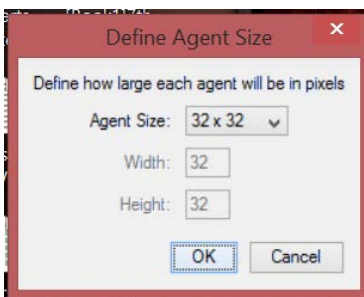
**STUDENT HANDOUT 1A Provides step-by-step instructions for creating a worksheet** *(Found on page 3 in the student packet)*

Step 1: Open the program. When AgentSheets first launches, all the students will see is the main tool bar which looks like this:



Click on FILE>>NEW PROJECT to create a new game.

Have each student create a new FROGGER game and instruct students where to save their game.



This box will appear. Students should just click OK.

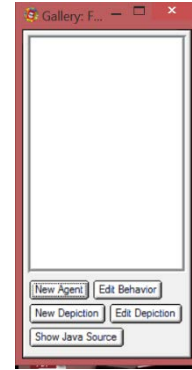
A gallery window will now appear. Let the students know this is how one creates agents in AgentSheets. Using the knowledge from the earlier discussion, ask the students what agents must be created.

Ask the students to create their Frog agent, by clicking on New Agent.

The student will be asked to name the agent – they should name it Frog.



A standard icon appears for the Frog. Have the students click Edit Depiction to change how the frog looks. (Be sure the students know the word “depiction”!).

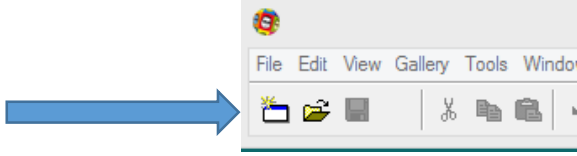
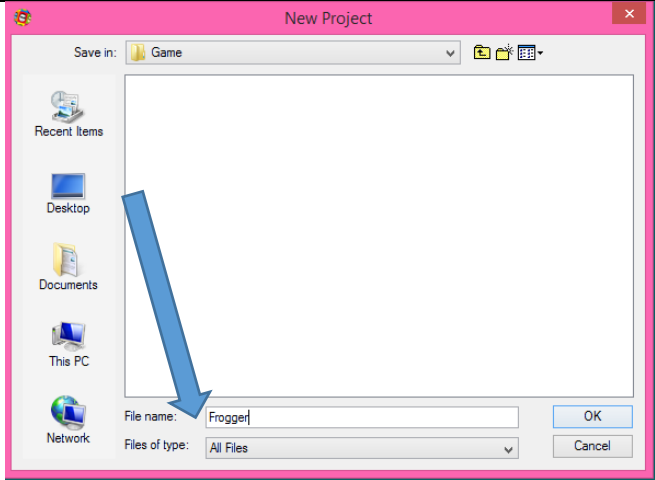
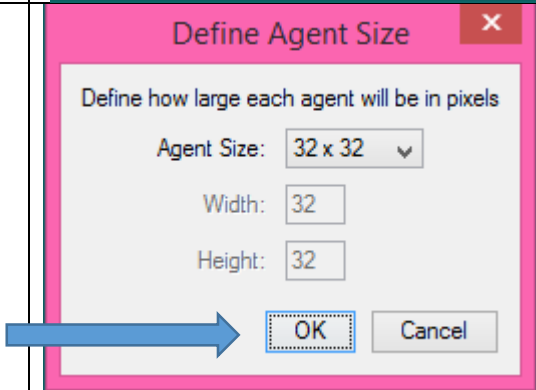


**Pass out Student Handout 1B to each student or group of students. This provides step-by-step instructions on making agents. (Found on page 4 in the student packet)**

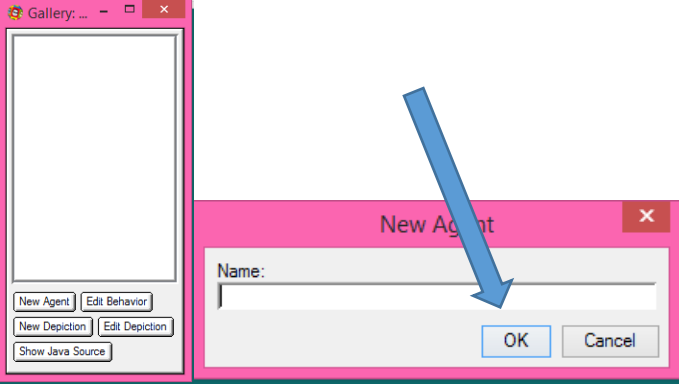
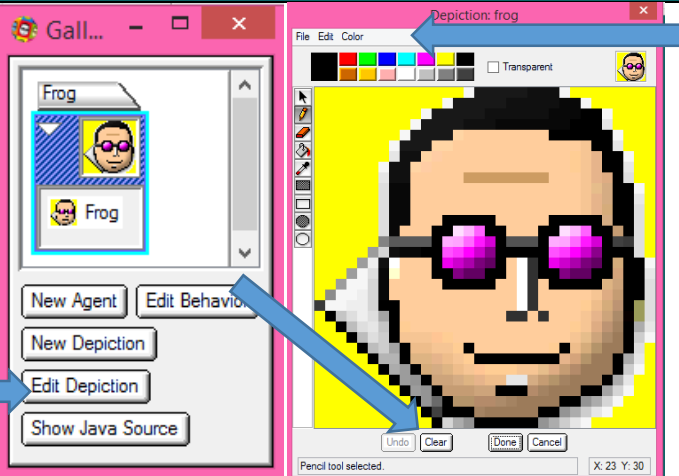
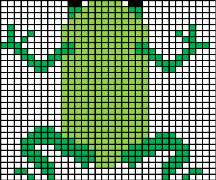
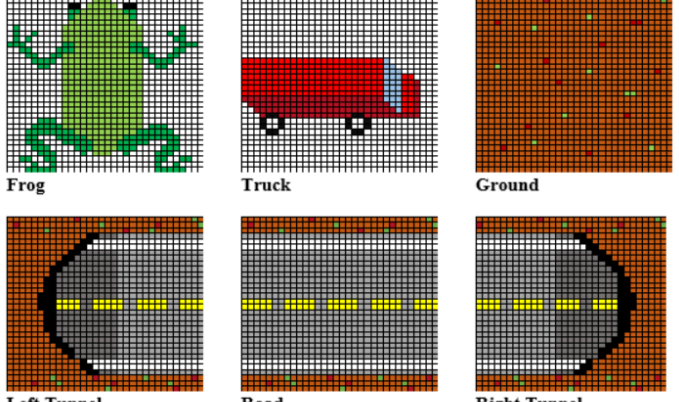
Once the Frog is made, students should continue on to make a street, a left tunnel, a right tunnel, ground, and a truck.

**Student Handout 1C shows possible designs for each agent, but encourage students to create their own designs if time allows. (Found on page 5 in the student packet)**

## Student Handout 1A: Create a game

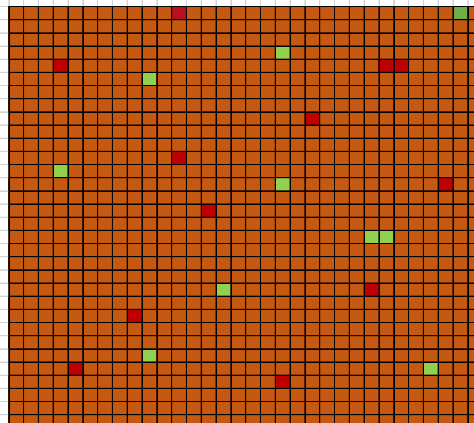
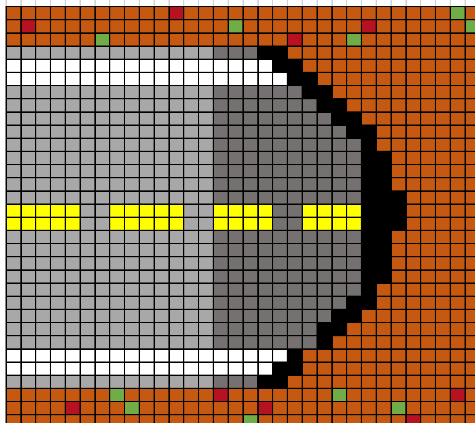
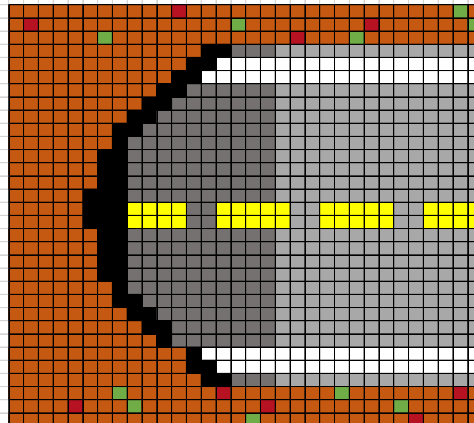
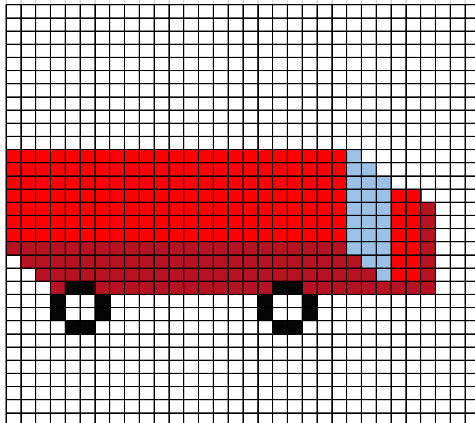
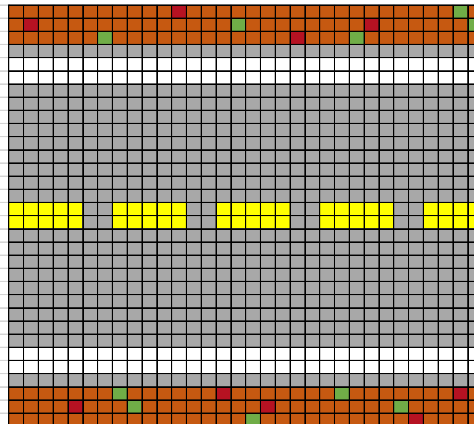
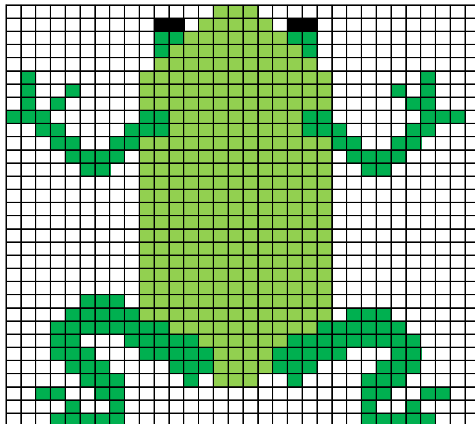
Step 1	<b>Create Game</b>  <b>Click on the new game icon (far left)</b>	
Step 2	<b>Name the Game</b>  <b>Name it Frogger and click OK</b>	
Step 3	<b>Define Agent Size</b>  <b>Do not change - Click OK</b>	

## Student Handout 1B: Create agents

<p><b>Step 4</b></p>	<p><b>Create Agent</b></p> <p>Click on New Agent</p> <p>Name it Frog</p> <p>Click ok</p>	
<p><b>Step 5</b></p>	<p><b>Edit Agent</b></p> <p>Click Edit Depiction</p> <p>Click Clear to erase the current image.</p>	 <p>Click on Color&gt; Mask Color&gt;&gt; White to make the white background sections see-through</p>
<p><b>Step 6</b></p>	<p><b>Draw Frog</b></p> <p>Click Done</p>	 <p>Here is an example of one way to draw the frog. You can be creative. If you make a mistake, use the eraser or click CLEAR to clear the whole area.</p>
<p><b>Step 7</b></p>	<p><b>Draw remaining agents</b></p>	<p>Agents:</p>  <p>Frog      Truck      Ground</p> <p>Left Tunnel      Road      Right Tunnel</p>

## Student Handout 1C: Agent Creation Models

Use these as quick starting points for your own agent. They don't have to look exactly like the model!

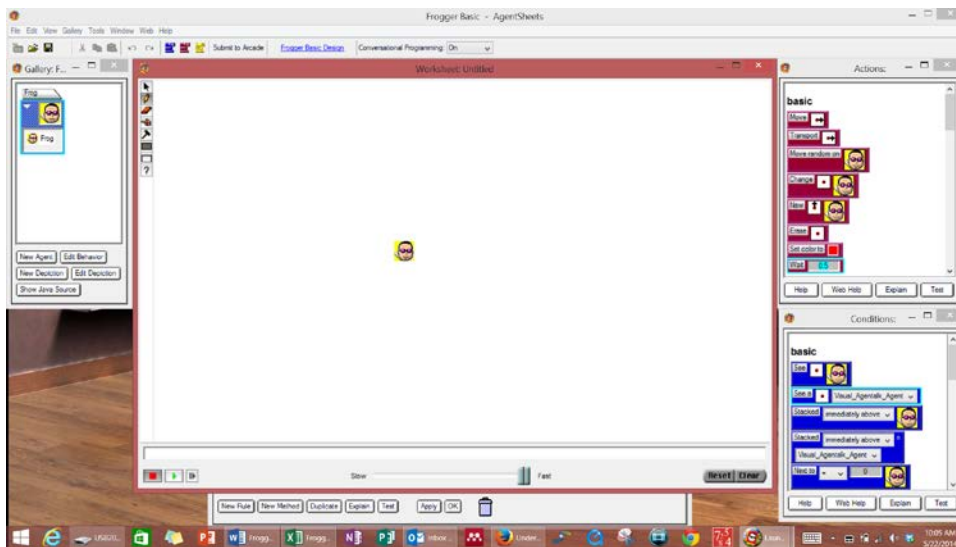


## Teacher Instructions:

### Part 2 – Create the Worksheet for the game

Now the students will use their agents to make a worksheet. To do this, the student must click FILE>>NEW WORKSHEET

A new worksheet will appear. For ease of use, the students should make it bigger than the standard size, but it is easier if they do not make it full screen so they can still see other boxes such as the gallery box. Especially as students are learning, it is also helpful to make it a bit smaller, so that it can be quickly redrawn if mistakes are made!



Once the worksheet is open, the students can create their game space. Generally, it should have a bit of ground at the bottom for the frog to sit on before the game starts. Then it has a street, and then space above to later add the river.

It would be good to have a student sketch out what the worksheet might look like on the board. An example of a worksheet is shown in the student handouts.

**Pass out Student Handout 2 – Remind students to save their worksheet once it is created.** *(Found on page 6-7 in the student packet)*

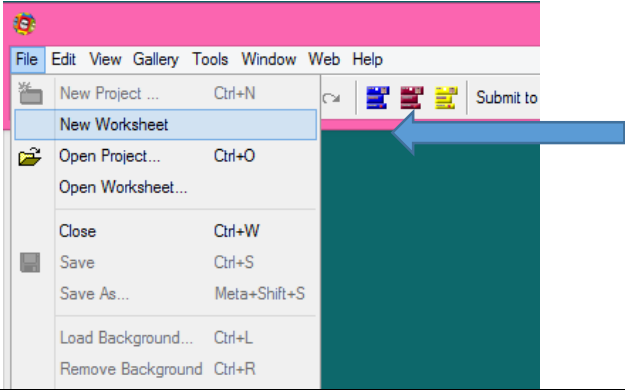
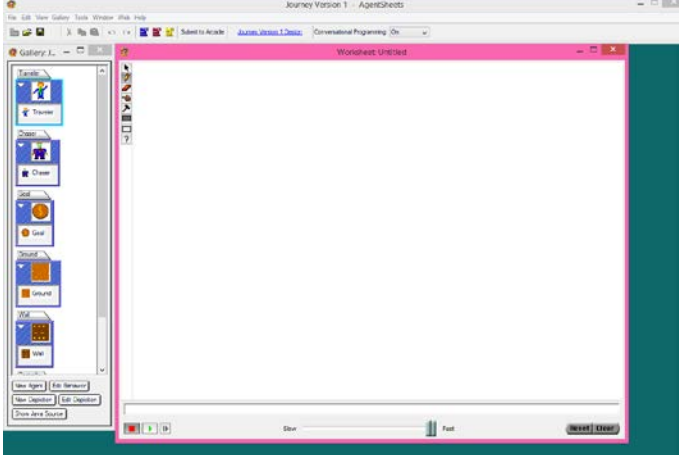
#### ABOVE

This word is used in two different ways when designing this game.

The grotto is above the river versus the frog is above (or on top of) the log. Mention this to your students as it will be important to choose their words carefully.

## Student Handout 2: Part 2 – Create Worksheet

The worksheet is the game space –  
it is where the agents will perform their actions.

<p><b>Step 8</b></p>	<p><b>Make the worksheet</b></p> <p>Click <b>File&gt;&gt;New Worksheet</b></p>	
<p><b>Step 9</b></p>	<p><b>Make the worksheet bigger</b></p> <p>Notice it is big, but not so big that it fills up the whole space.</p>	



**Select Tool**

**Pencil Tool** – places a single agent on the worksheet

**Eraser** – erases agents from the worksheet

Will be defined later

Will be defined later

**Draw Rectangle** – places agents in an array (rectangle)

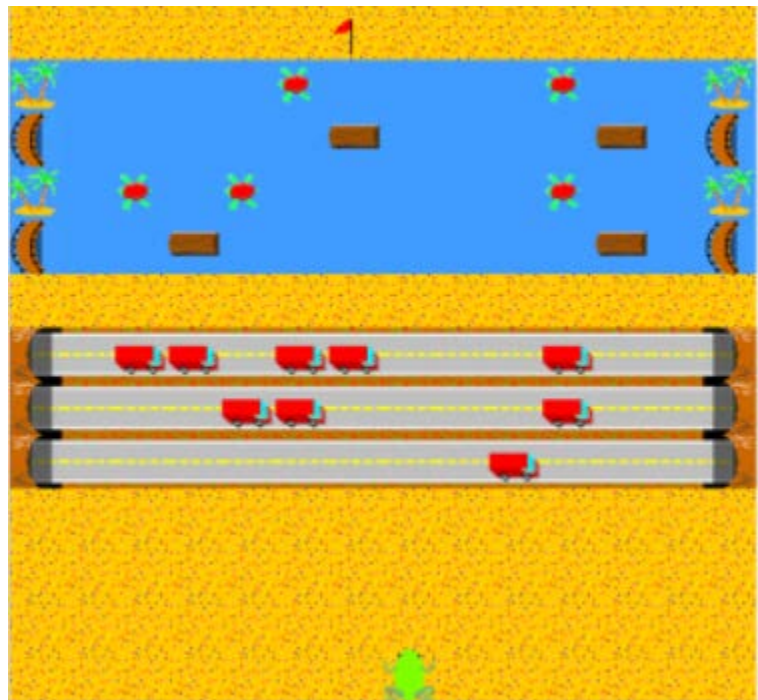
**Erase Rectangle** – erases agents in an array

Will be defined later



<p><b>Step 10</b></p>	<p>Use the tools to place items on the worksheet.</p> <p><b>Pencil:</b> places agents one at a time</p> <p><b>Filled in Rectangle:</b> Places agents in an array.</p>	<div data-bbox="592 210 722 346" data-label="Image"> </div> <p>It is important that you do not draw over the Frog with the Street agent.</p> <p>This means if you place a Frog on the worksheet, do not draw the Street over it without erasing the Frog first.</p>
-----------------------	---	---

You should have only this much of your worksheet filled in for now!



**This is a good time to save the worksheet!**

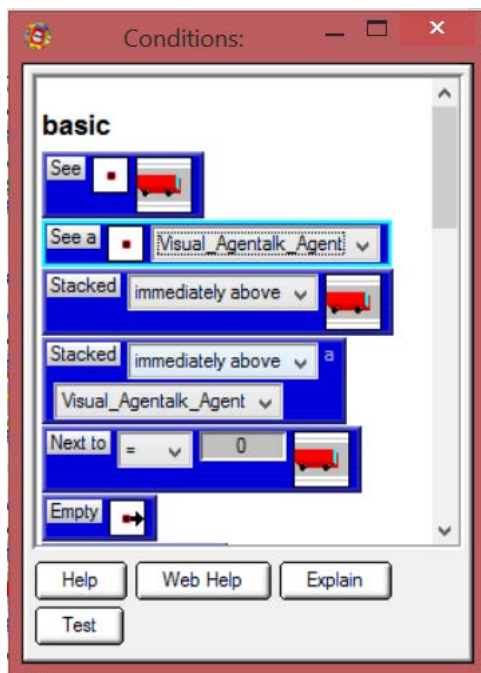


## Teacher Instructions:

### Part 3 – Understanding Agent behaviors

At this point, students should have all six agents created and their worksheet created. Now they are ready to give their agents behaviors.

Behaviors are thought of in this way...



IF...I have enough money.... THEN .... I can go to the baseball game.

Sometimes there is more than one condition

IF...I have enough money and I have a ride.... THEN .... I can go to the baseball game.

Sometimes there is more than one action

IF...I have enough money and I have a ride.... THEN .... I can go to the baseball game and buy a soda.

The CONDITIONS box lists the conditions under which the action will occur. The ACTIONS box lists the actions which occur when the CONDITIONS are true. **Student Handout 3a** is designed to give students practice in this. You can give it to them to solve in pairs, or use it as a class activity.

**Student Handout 3** details all of the behaviors for the agents. Consider whether your students need this support. Many students will be able to handle this programming without support once they have thought through the actions. *(Found on page 8 in the student packet)*

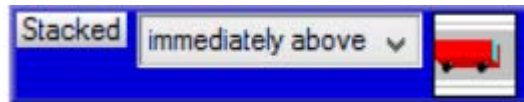
## Student Handout 3: Understanding Conditions and Actions

Explain each condition or action below

Conditions:



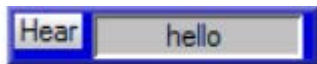
A \_\_\_\_\_



B \_\_\_\_\_



C \_\_\_\_\_



D \_\_\_\_\_



E \_\_\_\_\_

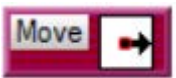


F \_\_\_\_\_



G \_\_\_\_\_

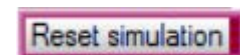
Actions:



A \_\_\_\_\_



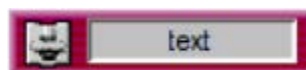
B \_\_\_\_\_



C \_\_\_\_\_



D \_\_\_\_\_



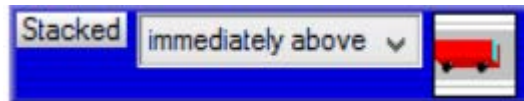
E \_\_\_\_\_

## Student Handout 3: ANSWER KEY

### Understanding Conditions and Actions

Explain each condition or action below

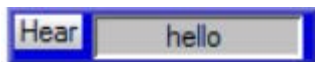
Conditions:



A IF the agent sees a truck to the right

B IF the agent is on top of the truck

C IF there is nothing to the right



D If the agent hears HELLO

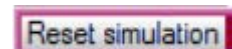
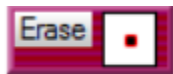
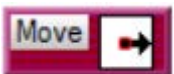
E If the user presses the A key



F If the value is equal to 0

G If the agent is next to less than or equal to five trucks

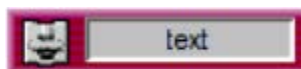
Actions:



A Move to the right

B Erase the agent

C Reset the simulation



D Change the agent to this image

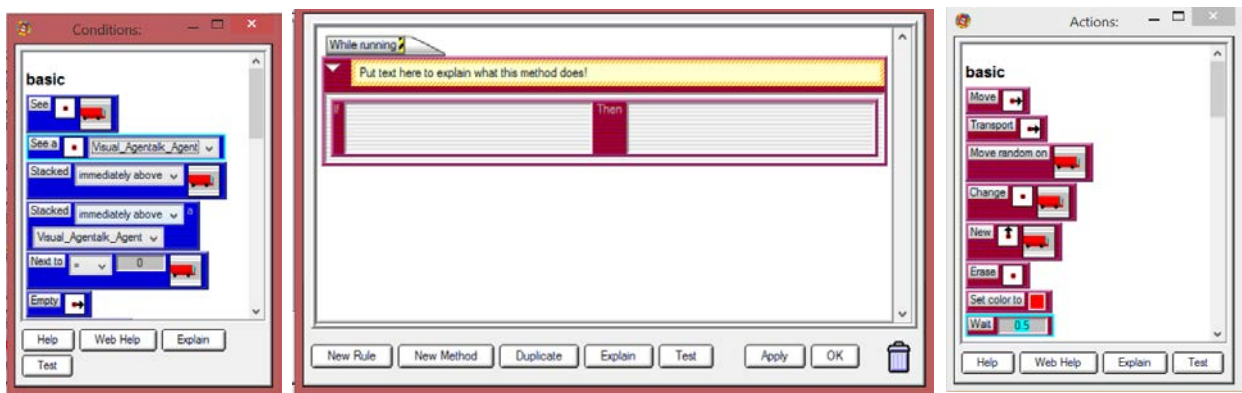
E Speak this text

## Teacher Instructions:

### Part 4 – Frog Behavior

Ask the students to edit behaviors of the Frog. To do this, they will click on EDIT BEHAVIOR on the FROG located in the GALLERY BOX.

Three boxes will appear as shown below: the CONDITIONS box, the ACTIONS box and the BEHAVIOR box. (It is helpful for the students to lay them out in this order)



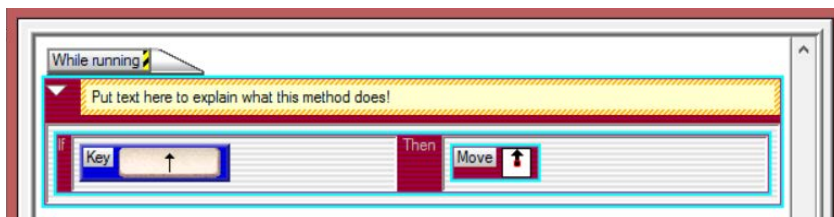
Use drop and drag to bring in conditions from the left side of the BEHAVIOR box, and actions from the right side.

Ask students...

How will we get the frog to move? Can we use voice control? (Just tell the frog to move?) No, what else could we use?

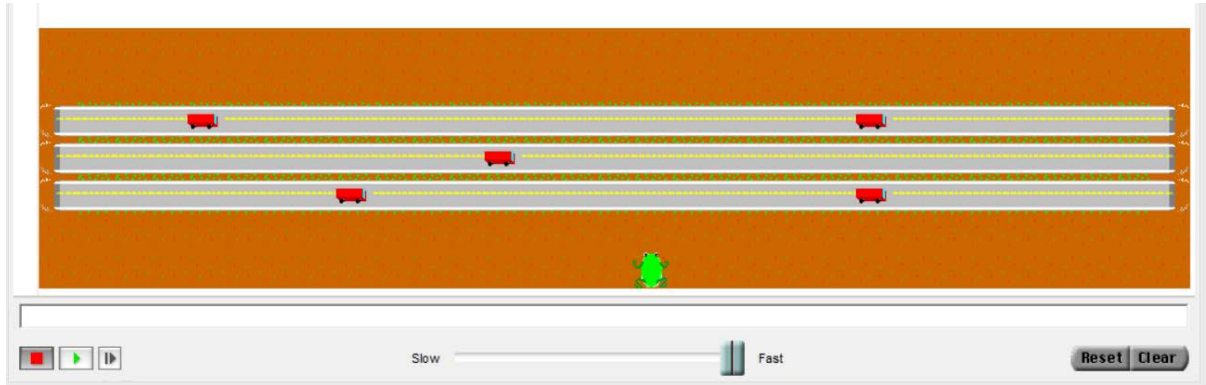
A student will likely suggest using the keyboard. Introduce the term CURSOR CONTROL such that you use the keyboard to move an Agent. Have students work together to determine the proper condition and action to make the frog move up.

After a couple of minutes of discussion, solicit ideas (correct code is shown below)

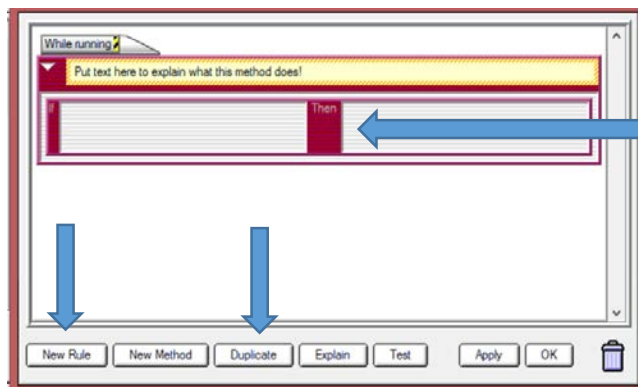


Press the UP ARROW key on the keyboard to get the up arrow indicator.

Have the students test their rule. Click on the green arrow (PLAY) to start the game. Have them press the up arrow to see if their frog moves up. Press the Red Square (STOP) to stop the game, and RESET to reset the worksheet. When the student presses the up arrow, the frog should move up.



Show students how to add rules by clicking NEW RULE at the bottom of the box. Also show them how to DUPLICATE rules using the DUPLICATE button.



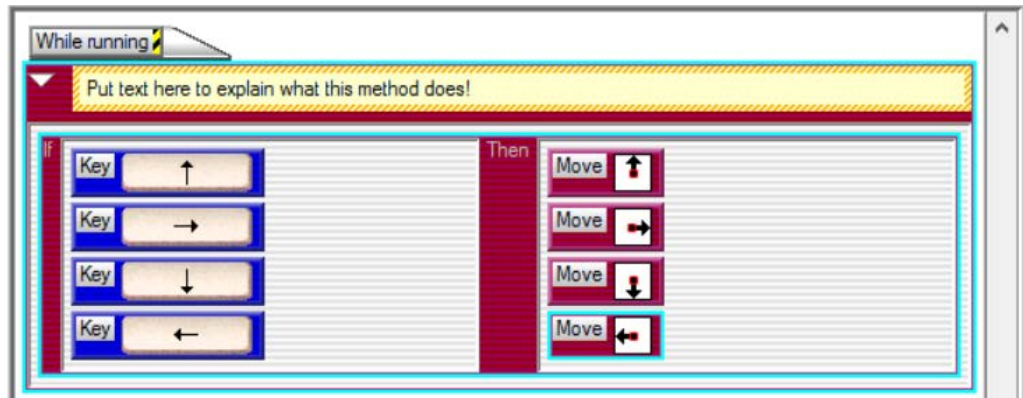
Select a rule by clicking in the center of it. Click on DUPLICATE to duplicate the rule. Click on DELETE to remove it.

Have the students create the rest of the rules to move their frog up, down, right and left.



**NOTE:** Some students will likely find out at this point that they didn't save their worksheet. If that is the case, they will need a few minutes to recreate their worksheet. You can avoid this by reminding everyone to save their worksheet before testing their game!

## COMMON MISTAKE:

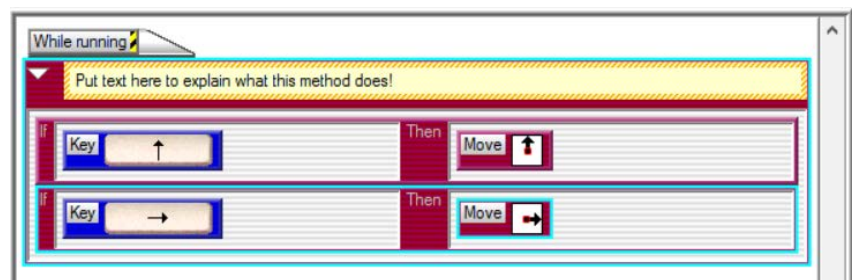


In this case, the agent is being ‘told’ ...if I click the up, down, right and left arrow, then you move up, down right and left.



*If students have this mistake, use it as a class discussion for them to determine why it’s wrong. Consider having them talk through (or act out) what it occurring.*

The rule should be,  
  
if I click the up arrow,  
you move up.  
  
if I click the right arrow,  
you move right.



They must each be their own, separate rule!

## Thoughts on Troubleshooting:

Think about ways to remove yourself as a teacher from being the only problem solver...

Consider these prompts when students ask for help...

- What have you already tried?
- Have you consulted with a friend?
- Have you considered whether your rule order is correct?

Resist the urge to find their mistake or take over their computer and fix their program!

Use the proper terms for the Computational Thinking Patterns (such as Cursor Control) so that students are comfortable with these terms.



## Teacher Instructions:

### Part 4 – Truck Behavior

Gather the class to talk about the trucks. Consider these prompts:

- What do the trucks do?
- Where do they come from?
- Where do they go?
- Which direction should they travel?
- How often do they travel on the street?
- What happens with the trucks when they reach the end of the street? Do the trucks disappear?
- Are more trucks created from the left? How are we going to get more trucks to appear?

#### Common Misconceptions:

Listen for students suggesting that the trucks somehow make a loop...that they go from the tunnel on the right 'back around' to the tunnel on the left.

This is a key conceptual misunderstanding.

By now your students should have a pretty good idea of what types of coding will be necessary. Solicit ideas and allow students to consider both correct and incorrect suggestions. Move students toward the idea that the tunnel on the left will GENERATE new trucks, while the tunnel on the right will ABSORB the trucks (erase them). With some discussion, students will move to the correct programming.

*The following instructions are provided to support the teacher – allow the students time to work on their own before providing assistance!*

#### Truck:



Explanation:

We want our Truck to move to the right every so often, as long as there is street to the right. Drag the "See" condition and click on its depiction parameter to define that the Truck needs to see a Street and its direction parameter (arrow) to indicate that it needs to see the Street to its right. Add the "Once every" condition and type 0.5 to specify that the truck will move

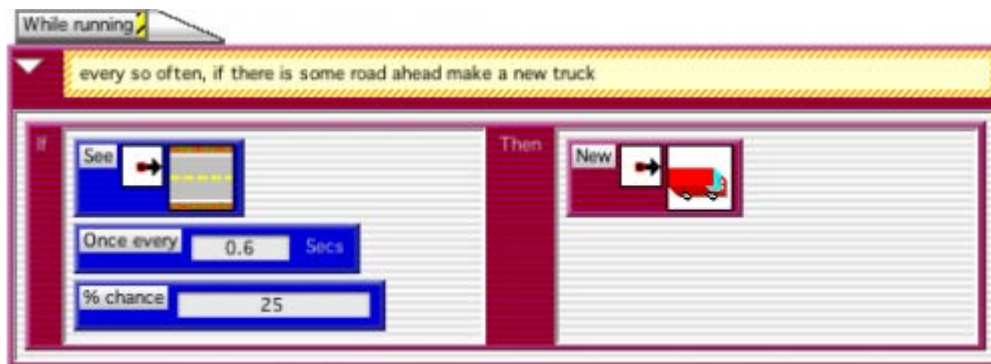
only every 0.5 seconds. Then, put the "move" action in the then part to complete the behavior.

We need to make an agent that GENERATES (creates) our truck on one side of the screen (because in Frogger, trucks appear on one side) and ABSORBS (erases) the truck on the other side (as the trucks "drive off" the screen). Think of this as a "tunnel" where the trucks originate from and disappear into.

## Add Truck-generation behavior to Tunnel agent:

We need a rule that says every once in a while, send out a new truck from the tunnel on the left.

Open the Behavior Editor for the Left Tunnel Agent and create a rule that looks like the following:



## Explanation:

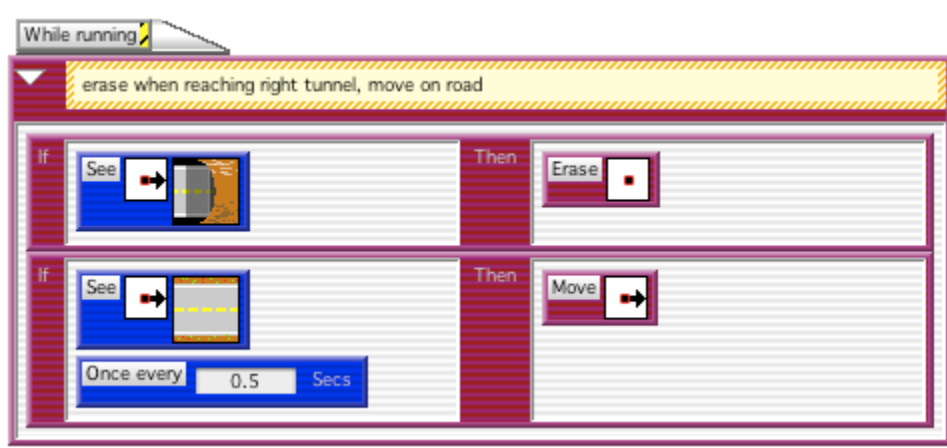
We want the Tunnel to create a Truck to its right, but only if there is street there. Otherwise, if there is, say another car, it shouldn't create one, because the Trucks will then pile up on top of each other. We also want our trucks to be created in a 'non-uniform' fashion, meaning we don't want the trucks to be created in such a way that the user can easily figure out the pattern of how often the trucks are created. So the behavior states "If there is street directly to the right of the Tunnel Agent, Then create a Truck once every 0.6 seconds." It also says that "this behavior has a 25% chance of happening." So every 0.6 seconds there's a 25% chance that the tunnel will create a truck agent on the street to the right.



## Add Truck Erase Behavior:

Now that we have the Tunnel at the end of the street, we need to tell the Truck to erase itself when it gets there (or, to think of it with Computational Thinking Patterns, to have the trucks ABSORBED by the tunnels).

Open the Truck's behavior editor and add a new rule by using the "New Rule" button at the bottom of the window. Then, add conditions and actions to have the Truck erase itself when it reaches the end tunnel. The complete behavior of the Truck with its two rules will look as follows:



Explanation:

If there is a tunnel to the right of the truck, the Truck should disappear by erasing itself (note that the "dot" in the direction parameter refers to the agent itself). So if there is a tunnel to the right of the Truck we have the Truck agent erased. Otherwise, if there is a street directly to the right of the Truck, the Truck will move to the right once every 0.5 seconds.

## Troubleshooting:

Now let's test our program to see how the Truck and Tunnel agents work. In the Worksheet window hit the run button.

Do the Tunnels on the left (at the beginning of the street) create Trucks?

Do the Trucks disappear when they reach the Tunnels on the right (at the end of the street)?

If one of these does not work, go back to the Truck and Tunnel behaviors and see if the behaviors are programmed correctly. If both of these work correctly play around with the game. Move the frog into the street and see if you can make it across. Try to have the frog get squished by a truck.

What happens? What's wrong?

## Teacher Instructions:

### Part 4 – Truck/Frog Collision

We're almost done but we're missing one extremely important behavior: the behavior that deals with the collision between the truck and the frog!

**Add Squished-Frog depiction:** Before we add the Frog-Truck collision behavior, we need to add a depiction that represents the squished frog to use when the frog gets hit by the truck. Please note that the squished frog isn't a new agent; it's just a squished version of our original frog agent. In the Gallery Window select the Frog agent. At the bottom of the Gallery Window click the "New Depiction" button and name it "Squished frog." Then, select the new depiction, and edit it to represent a squished frog.

**Add Frog-Truck Collision in the Frog's behavior:** Finally, we need to add another rule to the Frog. If our Frog encounters a Truck to its left, it needs to become a squished frog (basically if the frog is in front of the moving truck it gets squished). To accomplish this, we add the following new rule to the Frog agent behavior.



Explanation:

IF the Frog Sees a truck directly to its left,

THEN we first play the "honk" sound

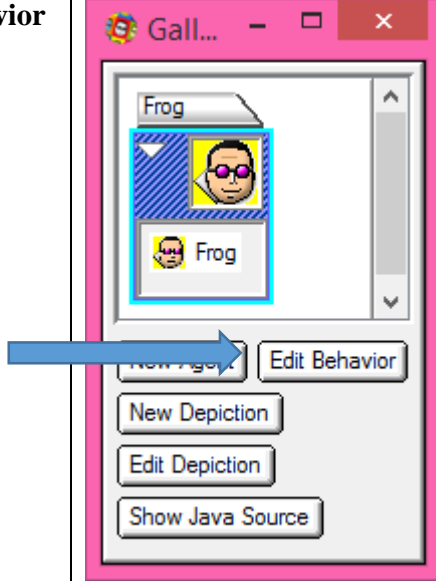

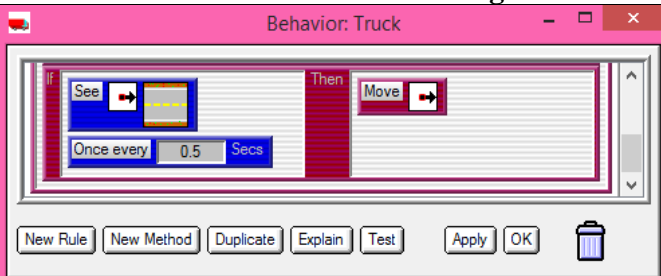
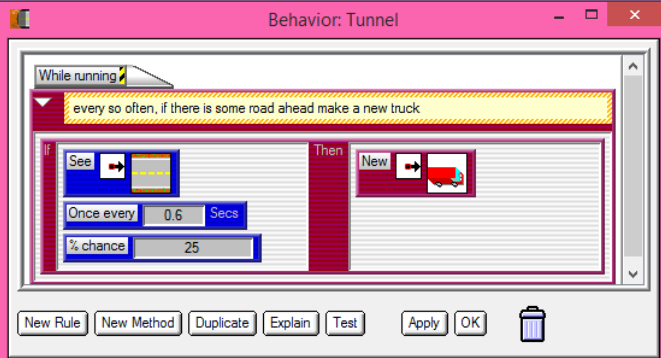
THEN we change the Frog's appearance to look like the squished frog.

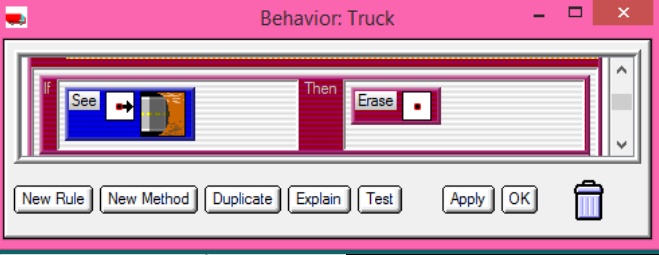
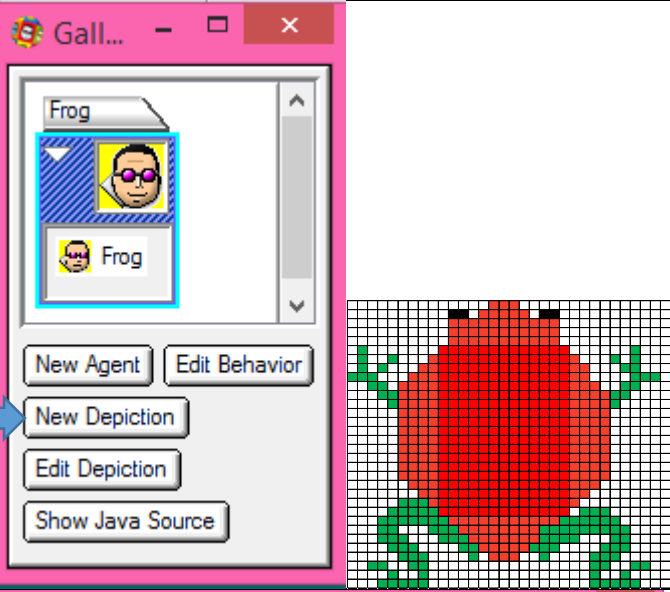
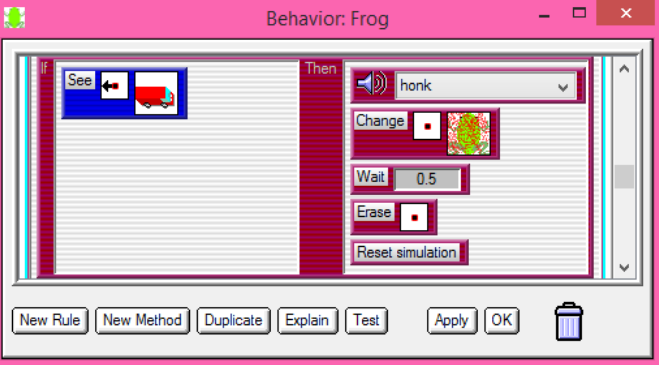
THEN we wait for 0.5 seconds (so that the player has enough time to see the squished frog and realize that the frog collided with a truck).

THEN, we erase our Frog agent and Reset the game so that the player can try again!

## Student Handout 4: Creating Agent Behaviors

*Click on the agent to add behaviors to that agent*

Step 1	Create behavior for the frog	
Step 2:	Cursor Control for <u>F</u> rog	 <p>This makes the Frog move UP when you push the UP arrow. Create the rest of the rules for the frog</p>
Step 3:	Make the trucks move <u>Click on the Truck Agent.</u> Add this behavior	
Step 4:	Make the <u>left tunnel</u> generate trucks	

Step 5:	Make the right tunnel <b>ABSORB</b> the <u>trucks</u>	
Step 6:	Create a 'squished frog' depiction	
Step 7:	Honk a horn and erase the <u>frog</u> when it collides with the truck	

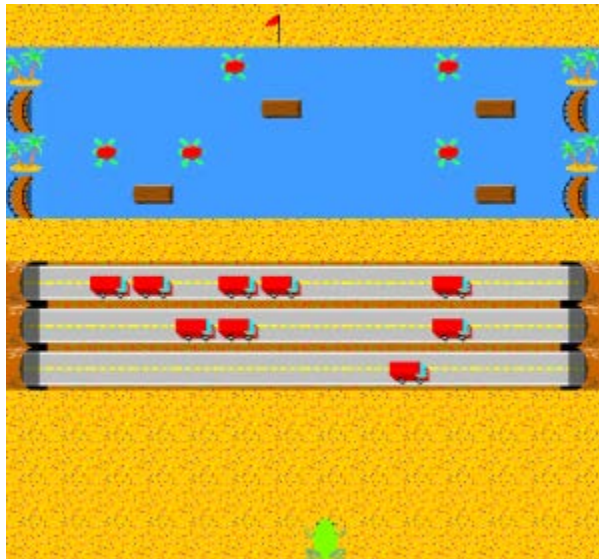
**You are ready to move on once the following items work correctly...**

- Does the frog move all directions?
- Do the trucks get generated (created) and absorbed (erased)?
- Does the Frog-Truck collision work correctly?

## Teacher Instructions:

### Part 5 – The River

Ask the students what is missing from their Frogger game... They should identify that they need a river with logs and turtles.



#### ACADEMIC LANGUAGE and COMPUTATIONAL THINKING PATTERNS

It is important to use the proper terminology during this discussion. As students use words like create, make, erase, disappear...restate their ideas using the terms GENERATE and ABSORB.

- River
- Logs -- We will have these move from left to right.
- Log-Maker (The bridges to the left of the river GENERATE (create) logs and the bridges to the right of the river ABSORB (erase) logs. This is similar to how the Tunnels to the left and right GENERATED and ABOSORBED trucks)
- Turtles -- We will have these move from right to left.
- Turtle-Maker (The Islands GENERATE and ABSORB turtles)
- Grotto (the flag at the top) so the Frog can beat the game

**Connect the actions of these agents back to the earlier parts of Frogger... Be sure students see the connections before moving on.**

- **River:** like the street
- **Logs/Turtles:** like the trucks
- **Log Maker (Bridge) and Turtle Maker (Island):** like the tunnels
- **Grotto (flag):** new

## Added Rules of the game:

**Have a class discussion to determine these rules... include discussions on how students might program these aspects.**

### Agent: River

- The Frog must drown if it falls in the river.

### Agent: Log

- Float On Water. We'll have the logs float from left to right
- Frog Must Be Able to hop on top of the Log
- Logs Must Disappear when it reaches the end of the river

### Agent: Log\_Maker (Bridge)

- Creates Logs if there is water to the right (Logs Float from Left to Right)

### Agent: Turtle

- Float On Water. Unlike the Logs, we'll have the Turtles float from right to left.
- Frog Must Be Able to hop on top of the Turtle
- Turtles Must Disappear when it reaches the end of the river

### Agent: Turtle\_Maker (Island)

- Creates Turtles if there is water to the left (because we want our Turtles to go from right to left).

### Agent: Frog :We must update the Frog Agent

- Jump on top of and move with the Logs and the Turtles
- The player loses if the frog falls in the water (the Frog Drowns)

### Agent: Grotto

- If the Frog gets to the grotto, the player wins!

**Provide students with Student Handout 5** *(Found on page 11-12 in the student packet)*

## Student Handout 5: The River

**You are tasked with creating the river scene of Frogger. Here are the rules:**

### Agent: River

- The Frog must drown if it falls in the river.

### Agent: Log

- Float On Water. We'll have the logs float from left to right
- Logs must TRANSPORT frogs
- Logs Must Disappear when it reaches the end of the river

### Agent: Log\_Maker (Bridge)

- GENERATES Logs if there is water to the right (Logs Float from Left to Right)

### Agent: Turtle

- Float On Water. Unlike the Logs, we'll have the Turtles float from right to left.
- Frog Must Be Able to hop on top of the Turtle
- Turtles Must Disappear when it reaches the end of the river

### Agent: Turtle\_Maker (Island)

- Creates Turtles if there is water to the left (because we want our Turtles to go from right to left).

### Agent: Frog :We must update the Frog Agent

- Jump on top of and move with the Logs and the Turtles
- The player loses if the frog falls in the water (the Frog Drowns)

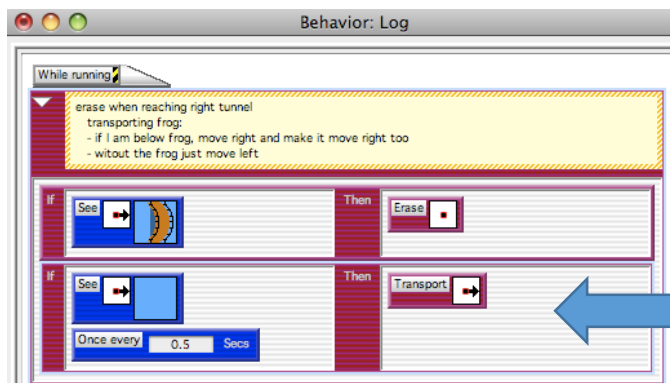
### Agent: Grotto

- If the Frog gets to the grotto, the player wins!

**Step 1: Create missing agents (river, log, bridge, turtle, island, grotto) and add them to the worksheet.**

**Step 2: Program the bridge to generate logs. Program logs to disappear when they reach the end of the water.**

**Step 3: Program the log to float down the river, from left to right.**



**Notice that we used TRANSPORT, not MOVE. This is so that the log can carry a frog!**

**Step 4: Test the program. You are ready to move on when you can answer YES to these questions:**

- Do Logs get created?
- Do the Logs Move across the river and disappear when they reach the Log Maker Agent?
- Does the Frog Get Transported when it jumps on the log?

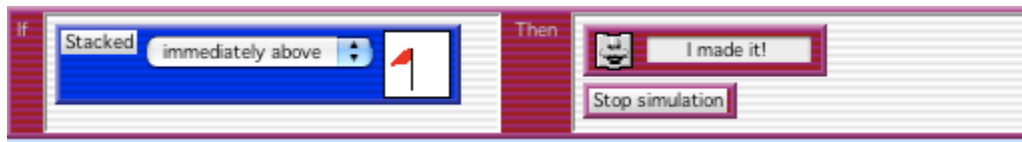
**Step 5: Program the island to generate turtles. Program turtles to disappear when they reach the end of the water.**

**Make sure the turtles float from RIGHT to LEFT.**

**Step 6: Test your program. You are ready to move on when you can answer YES to these questions:**

- Do Turtles get created?
- Do the Turtles Move and disappear when they reach the Turtle Maker Agent?
- Now, Control the frog and try to jump on a turtle, What happens? Does the Frog Move with the Turtle?

**Step 7: Program the game so that you win when the frog reaches the flag. Give the frog this behavior:**



**Step 8: Test your program. You are ready to move on when you can answer YES to these questions:**

- Does a message get played or appear when the frog reaches the Grotto?
- Does everything else work like you expect it to?



## End of Unit Review Sheet – Frogger

*(Found on page 13 in the student packet)*

A) The main computational thinking patterns we covered are:

- 1) **Cursor Control**: intentionally moving an agent.
  - a. Using keyboard keys to move an agent.
  - b. Example is moving the frog.
- 2) **Generate**: create new agents on the screen.
  - a. Use the “New” action in AgentSheets.
  - b. Examples are generating new trucks, turtles, logs in Frogger.
- 3) **Absorb**: deleting agents on the screen.
  - a. Use the “Erase” action in AgentSheets.
  - b. Examples are erasing the trucks, turtles, and logs on the other side of the screen.
- 4) **Transport**: transporting an agent along with another, as if one agent is carrying the other.
  - a. Use the “Transport” action in AgentSheets.
  - b. Examples are transporting the frog on the logs and turtles.
- 5) **Collision**: when 2 agents collide (run into each other).
  - a. Use the “See” condition
  - b. Use the “Stacked” condition, OR
  - c. Use the “Next to” condition.
  - d. Examples are the truck colliding with the frog.

B) Other concepts we covered in AgentSheets are:

- 1) Creating projects, worksheets, and agents.
- 2) Changing depictions for different circumstances, such as the ‘squished frog’ depiction.
- 3) Stopping and resetting the simulation.
- 4) Troubleshooting the simulation, and considering rule order.
- 5) Using sounds and messages in the game.
- 6) Timing our actions using the “Once every” condition.
- 7) Creating some random actions using the “% chance” condition, like when we wanted to generate trucks but not always to avoid too much traffic!
- 8) Creating comments or notes that explain what you are doing in the code. It helps you remember what the code does when you read it later in the future, or share the code with other users.

## Student Handout:

*(Found on page 14 in the student packet)*

## Challenge 1.0: Alligators

### Before your start this challenge:

You must have a complete basic Frogger game with a street and river. The Frog should die if it is hit by a truck or if it falls in the river.

### Design Challenge:

Frogs should be able to jump on the alligators back and travel on them just like they are logs. BUT...frogs should die if they are in FRONT of the alligator.

### Gamelet Design Activity:

Circle nouns to identify the agents and underline the verbs to identify actions associated with each agent. Mark adjectives to identify new shapes for an agent.

**Create new agent:** alligator

### Create agent behaviors:

- The frog can ride on the alligators back
- The frog dies if it runs into the mouth of the alligator



## Student Handout:

*(Found on page 15 in the student packet)*

## Challenge 2.0: Prevent Cheating

### Before your start this challenge:

You must have a complete basic Frogger game with a street and river. The Frog should die if it is hit by a truck or if it falls in the river.

### Design Challenge:

Create controls so that there is no cheating to win the game...

The frog should die if he rides all the way to the end of the water  
The frog should not be able to walk on tunnels or islands

### Gamelet Design Activity:

Circle nouns to identify the agents and underline the verbs to identify actions associated with each agent. Mark adjectives to identify new shapes for an agent.

### Create agent behaviors:

- The frog should die if it rides all the way to the end of the water
  - How can it do this? Which agents need new behaviors?
  - Will you use a new depiction for the frog?
- The frog should not be able to walk on tunnels or islands
  - How will you stop it from doing this?



## Student Handout:

*(Found on page 16 in the student packet)*

## Challenge 3.0: Dodge Cars and Trucks

### Before your start this challenge:

You must have a complete basic Frogger game with a street and river. The Frog should die if it is hit by a truck or if it falls in the river.

### Design Challenge:

Create controls so the frog must not only avoid the trucks, but also the cars going the other direction...

### New Agents:

Create a car agent

### Update Worksheet

- Create one (or two!) two lane street(s). Trucks should move to the right, cars move to the left.
- Save the new worksheet

### Update behaviors

- Cars are generated and absorbed (will you need new tunnels?)
- Cars move to the right
- Frogs are squished if hit by the cars
- Squished frog means the end of the game



## ISTE Standards<sup>3</sup> specific to the implementation of Frogger (Denoted with (★))

### Creativity and Innovation

*Students demonstrate creative thinking, construct knowledge, and develop innovative products and processes using technology. Students:*

#### **Apply existing knowledge to generate new ideas, products, or processes:**

- ★ Design and develop games
- Design and develop computational science models

#### **Create original works as a means of personal or group expression.**

- ★ Design original games
- Model your local environment, e.g., ecology, economy

#### **Use models and simulations to explore complete systems and issues.**

- Model scientific phenomena, e.g., predator / prey models
- Create visualizations

#### **Identify trends and forecast possibilities.**

- Build predictive computational science models, e.g., how the pine beetle destroys the Colorado pine forest
- Build live feeds to scientific web pages (e.g, weather information), process and visualize changing information

### Communication and Collaboration

*Students use digital media and environments to communicate and work collaboratively, including at a distance, to support individual learning and contribute to the learning of others. Students:*

#### **Interact, collaborate, and publish with peers, experts, or others employing a variety of digital environments and media:**

- ★ Students work in teams to build and publish their simulations as web pages containing java applets.

#### **Communicate information and ideas effectively to multiple audiences using a variety of media and formats.**

- Effectively combine interactive simulations, text, images in web pages

#### **Develop cultural understanding and global awareness by engaging with learners of other cultures.**

- ★ Students and teachers from the four culturally diverse regions interact with each other

#### **Contribute to project teams to produce original works or solve problems.**

---

<sup>3</sup> ISTE Standards for Students (ISTE Standards•S) are the “standards for evaluating the skills and knowledge students need to learn effectively and live productively in an increasingly global and digital world.” <http://www.iste.org/standards/standards-for-students>

- \* Define project roles and work collaboratively to produce games and simulations

## Research and Information Fluency

*Students apply digital tools to gather, evaluate, and use information. Students:*

### Plan strategies to guide inquiry.

Explore web sites and identify interesting connections

### Locate, organize, analyze, evaluate, synthesize, and ethically use information from a variety of sources and media.

Find relevant related web-based information, compute derivative information

### Evaluate and select information sources and digital tools based on the appropriateness to specific tasks.

Understand validity of information, e.g. Scientific journal information vs. Personal blogs

### Process data and report results.

Write programs to access numerical information, define functions to process data and create output based on voice or plotting to represent data.

## Critical Thinking, Problem Solving, and Decision Making

*Students use critical thinking skills to plan and conduct research, manage projects, solve problems, and make informed decisions using appropriate digital tools and resources. Students:*

### Identify and define authentic problems and significant questions for investigation.

Define research questions and explore approach of exploration

### Plan and manage activities to develop a solution or complete a project.

- \* Outline sequence of exploratory steps
- \* Experience complete bottom-up and top-down design processes
- \* Employ algorithmic thinking for creating programs to solve problems

### Collect and analyze data to identify solutions and/or make informed decisions.

Collect data as time series, e.g., collect group size of predator and prey, export time series to excel, explore various types of graph representations, e.g.,  $x(t)$ ,  $y(t)$  or scatter  $y=f(x)$

### Use multiple processes and diverse perspectives to explore alternative solutions.

- \* Experience and understand design trade-offs, e.g. Bottom-up vs. Top-down

## Digital Citizenship

*Students understand human, cultural, and societal issues related to technology and practice legal and ethical behavior. Students:*

### Advocate and practice safe, legal, and responsible use of information and technology.

- \* Learn how to use tools to locate resources, e.g., images with google image search, but understand copyright issues

### Exhibit a positive attitude toward using technology that supports collaboration, learning, and productivity.

- \* Stay in the flow, where design challenges match design skills

## Frogger (Continued)

- \* Experience success through scaffolded game design activities
- \* Mentor other students

### **Demonstrate personal responsibility for lifelong learning.**

- \* Explore options of going beyond expected learning goals

### **Exhibit leadership for digital citizenship.**

- \* In a collaborative setting become a responsible producer of content for diverse audiences

## **Technology Operations and Concepts**

*Students demonstrate a sound understanding of technology concepts, systems, and operations. Students:*

### **Understand and use technology systems.**

- \* Know how to organize files and folders, launch and use applications on various platforms

### **Select and use applications effectively and productively.**

Know how to orchestrate a set of applications to achieve goals, e.g., make game and simulations using Photoshop (art), AgentSheets (programming), and

- \* Excel (data analysis).

### **Troubleshoot systems and applications.**

- \* Debug games and simulations that are not working

### **Transfer current knowledge to learning of new technologies.**

- \* Reflect on fundamental skills at conceptual level. Explore different tools to achieve similar objectives.