Creating "3D-Frogger" With AgentCubes Online

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 & Jeffrey Bush, University of Colorado, School of Education. Adaptations using AgentCubes and AgentCubes Online made by Catharine Brand

This curriculum 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, Catharine 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.

3D-Frogger

Lesson Objective:

- To create a game of Frogger
- To master the basics in the use of AgentCubes 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 a basic world 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

3D-Frogger

Table of Contents

Vocabulary/Definitions

(Found on page 2 in the student packet)

Absorb	**one agent consumes an incoming agent, making it 'disappear'. 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).**
Action	**what an agent does if the given conditions are met**
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**
Shape	**an alternate image of the original agent. For example, the frog can have two shapes: 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 goal is located, which must be reached to win the game**
Transport	**when one agent moves and carries whatever agents are on top of it; the ability of an agent to be on top of, and move with, another agent**

General Teaching Strategies[1]

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.
- These lessons are also designed to give students positive experiences with and perceptions of computer science. Research shows that students turn away from high school and college computer science courses if they perceive it as boring, unrelated to what matters to them, and hard. We hope to change that by providing a fun, relevant and accessible computer science experience where they can personalize their experience to make computer science about them.
- Guided discovery is the central tenant of our curriculum. Direct instruction is sometimes used for aspects where students are learning the code for the first time; however, materials have been provided to ensure that students understand the programming concepts, as opposed to simply copying code.
- Whenever possible, students should try to come up with the steps on their own or in small groups, when differentiation and more structure is needed, we have more structured materials available.

[1] 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.

Google. (2014). Women Who Choose Computer Science — What Really Matters The Critical Role and Exposure. Retrieved from https://www.google.com/edu/resources/computerscience/research/

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.

Taylor, H. G., & Mounfield, L. C. (1994). Exploration of the Relationship between Prior Computing Experience and Gender on Success in College Computer Science. *Journal of Educational Computing Research*, *11*(4), 291–306.

- 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, THIS IS OK! A little frustration and moving at a slower pace is well worth the deeper conceptual understanding that comes with guided discovery.

Guided Discovery Process

- **Model the process** rather than just giving students the answer. As a teacher, focus on explanations and discussions of WHY something works or doesn't work and let the students figure out HOW to make it work.

- Building the game on your own, before trying it with your class will enable you to see which steps may challenge or confuse your students.

- Have students work through problems independently or in small groups. Ask directing questions or give helpful suggestions, but **provide only minimal assistance** and only when needed to overcome obstacles.

- **Group work is your friend**! 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 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 that some solutions might be more effective than others. Also group work lets them see that they are not alone and that others have similar and different questions, struggles, inspirations and perspectives.

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

- "What have you tried so far? Why didn't it work?" is a great way to start any troubleshooting discussion.

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.
- Encourage discussion and reflection on these Computational Thinking patterns. Small group or whole class discussion relating Computational Thinking patters to the outside world can be super productive.
- Remember that conceptual understanding takes time, and it may be necessary to review 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. Plus, it makes grading and reviewing games more fun for you.
- 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.
- Consider students who are ahead to the role of “code ambassadors” to walk around and help their peers with coding questions.
- 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 by having challenge activity 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[2]. 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.

[2] http://sgd.cs.colorado.edu/wiki/Scalable_Game_Design_wiki

Teacher Instructions:

Part 1 – Create a World 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

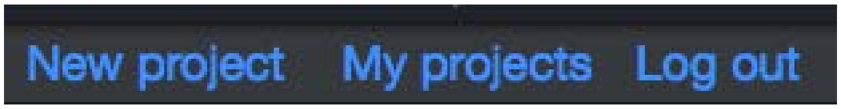
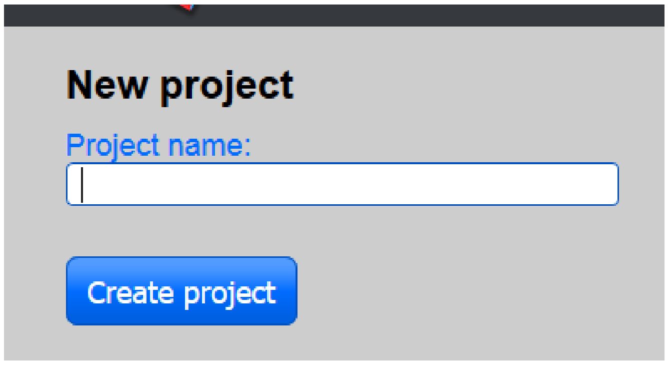
You are a frog. Your task is simple: hop across a busy highway, dodging trucks who come out of (and go back into) tunnels, until you finish crossing the river, where you must keep yourself from drowning by crossing safely to your grotto at the top of the screen by leaping across the backs of turtles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

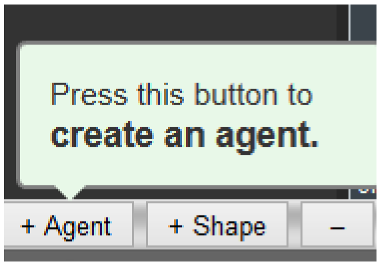
- *Trucks drive along the highway*
- *Trucks come out from the tunnels and go back into the tunnels*
- *Logs and turtles float along the river*
- *Snakes and alligators eat the frog*

Helpful Tips

This lesson will be teacher-directed in the beginning so that students learn AgentCubes Online while learning how to program. Be sure to give students time to work independently and figure things out using the program.

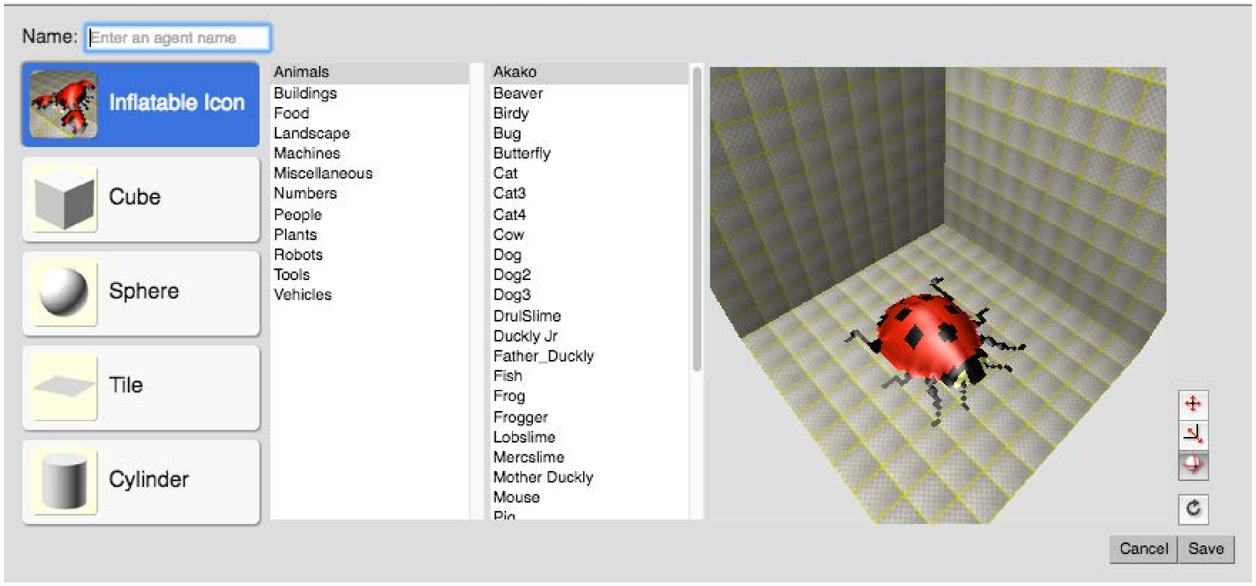
STUDENT HANDOUT 1A Provides step-by-step instructions for creating a World *(Found on page 3 in the student packet)*

Step 1: **Create Project.**	 **Go to https://www.agentcubesonline.com/** **If you have an account, click on the Login link. If not, click on the Sign up link.** **After you login, click on the blue "New project" link below your login name.**
Step 2: **Name the Game** **Name it Frogger and click Create project.**	

Step 3: Create Agent **Click on New Agent at the bottom left of the AgentCubes Online window.**	 Press this button to create an agent. + Agent + Shape –

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 the +Agent button. This window will appear:



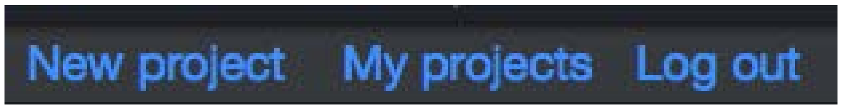
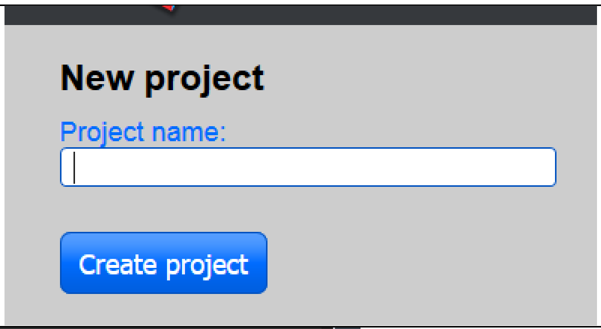
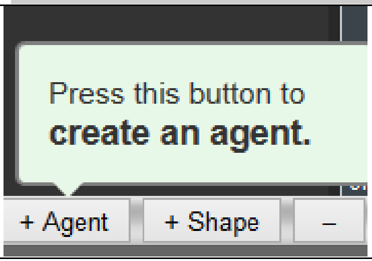
The “Frog” can be made from any of the premade agent depictions, They can make it “Dogger” or “Dragonger” if they want.

Student Handout 1B provides step-by-step instructions on making agents. *(Found on page 4 in the student packet)*

Instructions are provided to the student allowing them to edit their frog or create their own frog. Note that students can update their frog later in the game development.

Once the Frog is made, students should continue on to make a road, a left tunnel, a right tunnel, ground, a truck and a Grotto (flag or star).

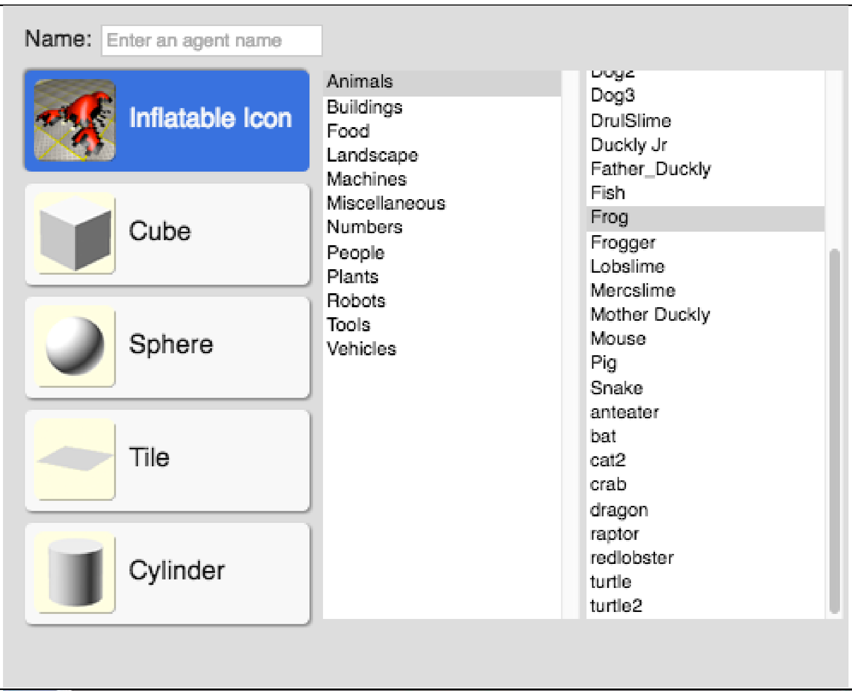
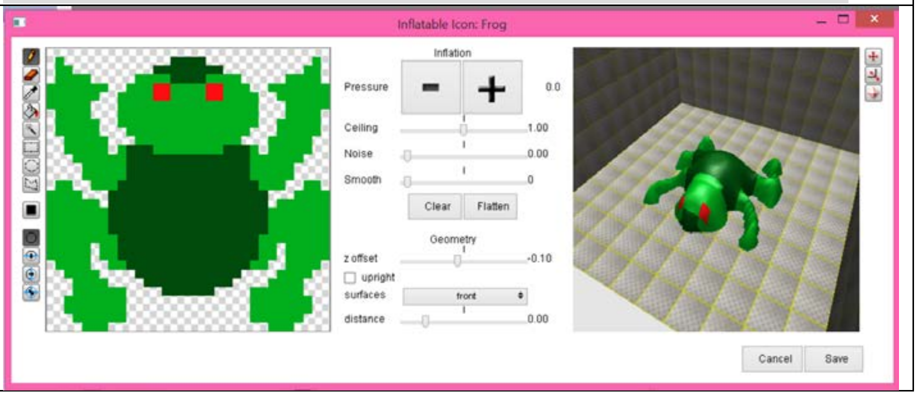
Student Handout 1A: Create a game

Step 1	**Create Game**	 **Go to https://www.agentcubesonline.com/** **If you have an account, click on the Login link. If not, click on the Sign up link.** **After you login, click on the blue "New project" link below your login name.**
Step 2	**Name the Game** **Name it Frogger and click OK**	
Step 3	**Create Agent** **Click on +Agent button in the lower left corner**	 Press this button to create an agent. + Agent + Shape –

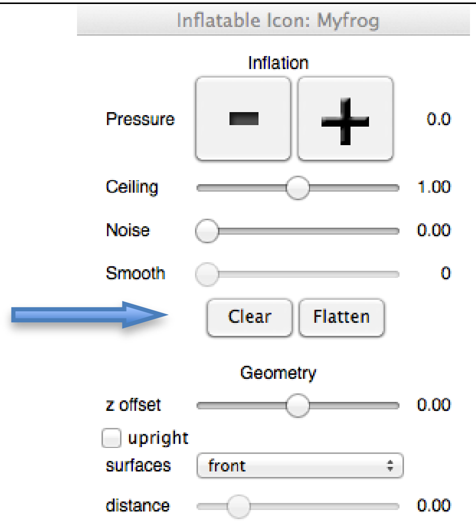
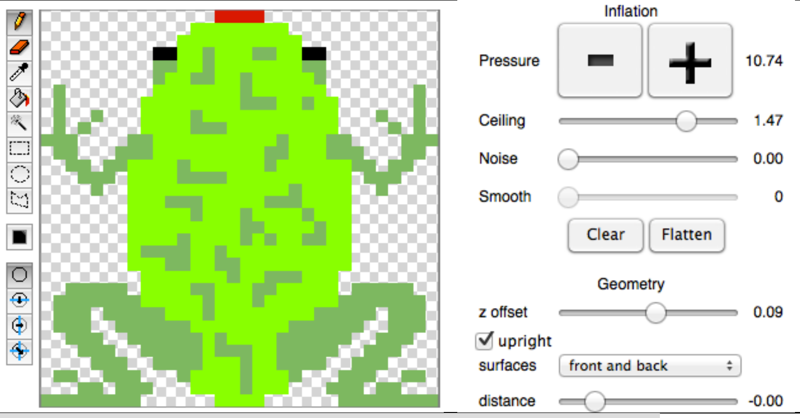
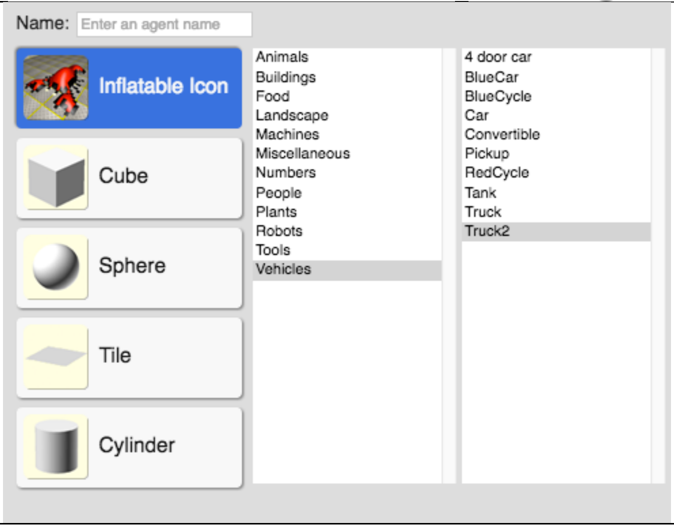
SCALABLE GAME DESIGN

3D-Frogger

Student Handout 1B: Create agents

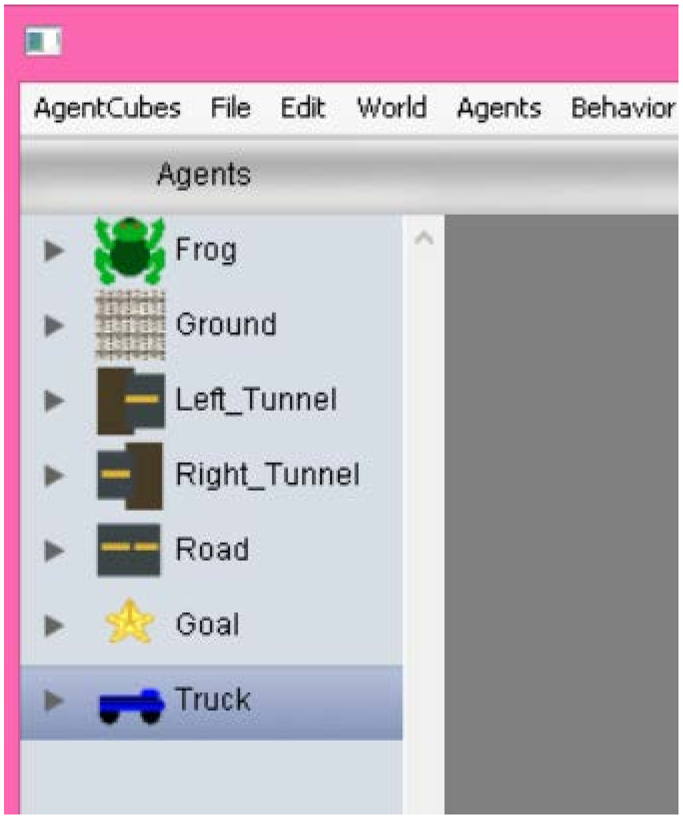
Step 4	**Make sure to give your agent a name** **Click OK to save your agent. It will appear on the left of your screen.**	
	You can double-click on the agent picture to edit how it looks. **This only works for inflatable icons,**	

3D-Frogger (Continued)

Step 5	**Use the pencil tool to add details to the frog.** **Or use the clear button to erase the frog.**	 Inflatable Icon: Myfrog Inflation Pressure 0.0 Ceiling 1.00 Noise 0.00 Smooth 0 Clear Flatten Geometry z offset 0.00 ☐ upright surfaces front distance 0.00
Step 6	**Click on color well to choose a color.** **Use the + button to inflate your drawing and make it 3D.** **Click Save**	 Inflation Pressure 10.74 Ceiling 1.47 Noise 0.00 Smooth 0 Clear Flatten Geometry z offset 0.09 ☑ upright surfaces front and back distance -0.00
Step 7	**Create Truck, Left and Right Tunnels, street, ground, and a Grotto or Goal** **These are found in the vehicle and landscape categories of "inflatable icon"**	 Name: Enter an agent name Inflatable Icon, Cube, Sphere, Tile, Cylinder Animals, Buildings, Food, Landscape, Machines, Miscellaneous, Numbers, People, Plants, Robots, Tools, Vehicles 4 door car, BlueCar, BlueCycle, Car, Convertible, Pickup, RedCycle, Tank, Truck, Truck2

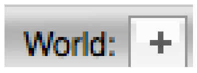
You are ready to move on when you have the following agents created:

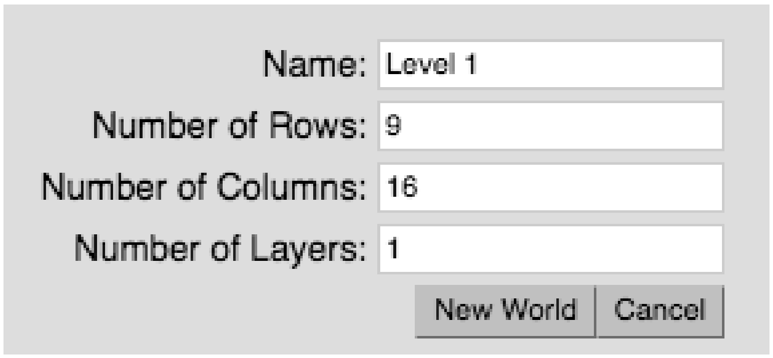
- ☐ **Frog**
- ☐ **Truck**
- ☐ **Left tunnel**
- ☐ **Right tunnel**
- ☐ **Grotto/Goal**
- ☐ **Street**
- ☐ **Ground**



Teacher Instructions:

Part 2 – Create a World for the game

Now the students will make a world. To do this, the student must click on the + next to the word "World" in the top bar of the AgentCubes Online window. 

This box will appear: 

Name:	Level 1
Number of Rows:	9
Number of Columns:	16
Number of Layers:	1

Have the students name their world "Level 1" and click OK. They should **not change the numbers** (which control the size of the world) this time.

Once the world is created, the students can design their game space by **using the pencil tool to draw agents** on it. This world should have 1 row of ground agents at the bottom for the frog to sit on before the game starts. Then it has 3 rows of street agents, with space above to later add the river.

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

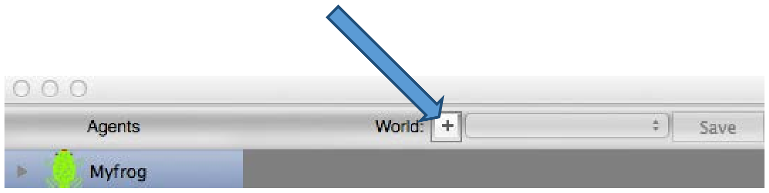
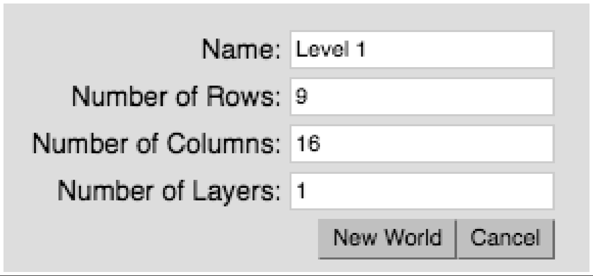
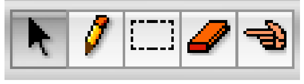
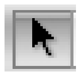
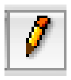
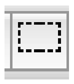
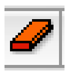
ABOVE
This word is used in two different ways when designing this game. The grotto is above the river versus the frog is above (or on top of) the log. Mention this to your students as it will be important to choose their words carefully.

Remind students to save their World when they have drawn the agents on it. ***(Found on page 7-8 in the student packet)***

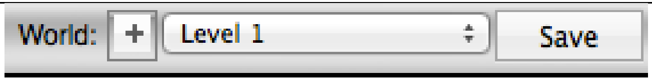
Student Handout 2: Part 2 – Create a World

The world is the game space –

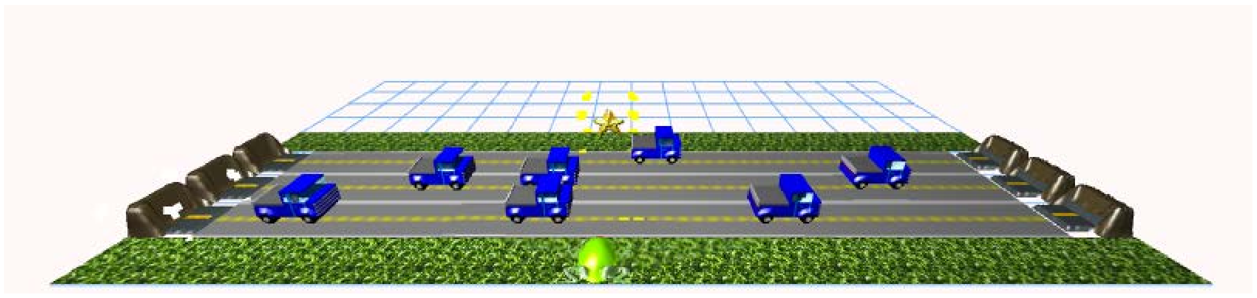
it is where the agents will perform their actions.

Step 8	**Create a world** **Click the + next to World in the top bar of the AgentCubes Online window**	
Step 9	**Name Your New World "Level 1"** **Do not change the numbers for your first world** **Click OK**	Name: Level 1 Number of Rows: 9 Number of Columns: 16 Number of Layers: 1
Step 10	**Try out the World tools** **Select tool for moving agents** **Pencil tool for drawing agents on World** **Tile tool for drawing groups of agents** **Eraser tool**	

Step 11	**Use these tools to place the agents on the World and make the street scene.** **Remember to place agents that**	**It is important that you do not draw over the Frog with the Street agent.** **This means you should start with the agents that go on the bottom like the ground and river, if you accidentally placed an upper agent first, you can just delete it and re-place it.**

	go on the bottom first	
Step 12	**Try out the 3D movement tools** **Rotate** **Pan: Move the world up and down.** **Zoom** **Move your World so that you can see the 3D shapes.**	
Step 13	**Use the <u>Save</u> button next Level 1 to save your World when you like the way it looks!**	 **Only save your World when it is set up to start a game.** **Then you can use the Reset button** **to return to the saved starting point for your game.**

Your World might look like this:

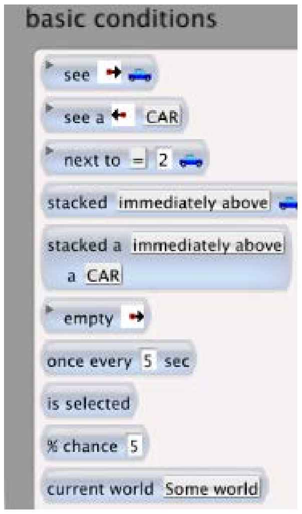


Teacher Instructions:

Part 3 – Understanding Agent behaviors (*Found on page 9)*

At this point, students should have all six agents created and their level 1 world created. Now they are ready to give their agents behaviors.

Behaviors are thought of in this way…



IF…I dinosaurs were not extinct ….   THEN …. I would get a pet T-Rex.

Sometimes there is more than one condition

IF…I dinosaurs were not extinct AND could be trained….   THEN …. I would get a pet T-Rex.

Sometimes there is more than one action

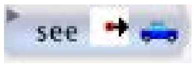
IF…I dinosaurs were not extinct AND could be trained….   THEN …. I would get a pet T-Rex AND teach it to arm wrestle.

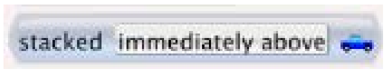
The CONDITIONS box lists the conditions under which the action will occur. The ACTIONS box lists the actions that occur when the CONDITIONS are true. **Student Handout 3** is designed to give students practice in this. You can give it to them to solve in pairs, or use it as a whole-class activity.

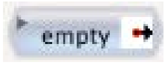
Student Handout 3: Understanding Conditions and Actions

Explain each condition or action below

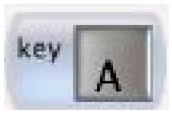
Conditions:

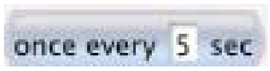




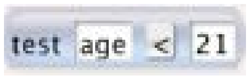


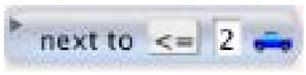
A__________________ **B**______________________________ **C**___________





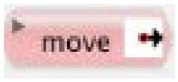
D_________________________ **E**________________________

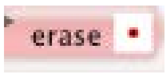


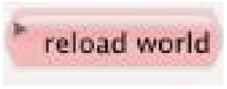


F_________________________ **G**________________________

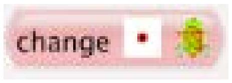
Actions:

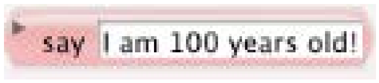






A__________________ **B**_________________________ **C**__________________



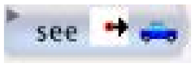


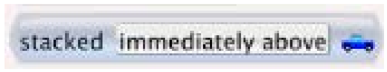
D_________________________ **E**________________________

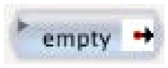
Student Handout 3: ANSWER KEY
Understanding Conditions and Actions

Explain each condition or action below

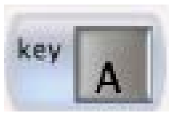
Conditions:

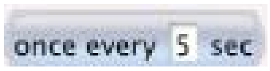




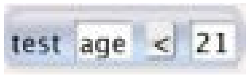


A IF the agent sees a car to the right **B IF the agent is on top of the car** **C IF there is nothing to the right**

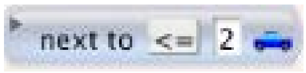




D If the player presses the A key **E Every 5 seconds**

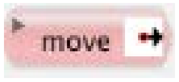


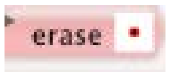
F If age is less than 21

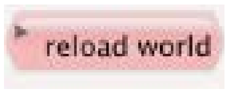


G If the agent is next to less than or equal to two cars

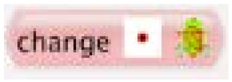
Actions:

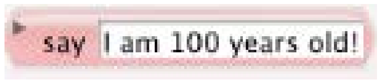






AMove to the right **B Erase the agent** **C Reload the saved World**





D Change the agent to this image **E Say "I am 100 years old!"**

Student Handout 4 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 10 in the student packet)***

Teacher Instructions:

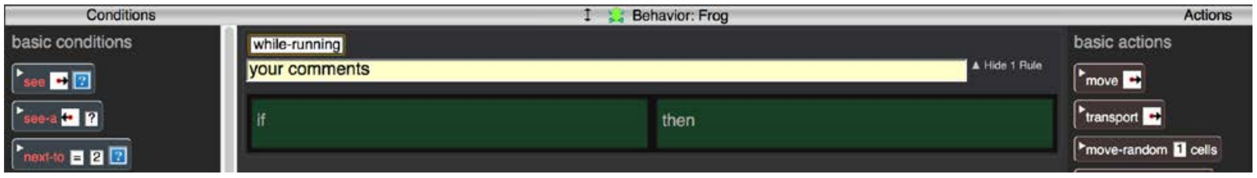
Part 4a – Frog Behavior

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?

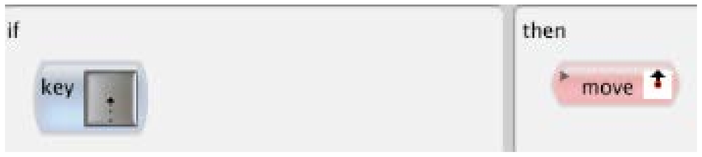
A student will likely suggest using the keyboard. Introduce the term CURSOR CONTROL such that you use the keyboard to move an Agent. Have students work together to determine the proper condition and action to make the frog move up.

Ask the students to edit behaviors of the Frog. To do this, they will click on the Frog in the list of agents. The Frog Behavior window will appear between the Conditions and the Actions palettes.

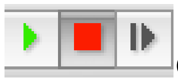
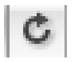


Use drop and drag to bring in conditions from the left side of the BEHAVIOR box, and actions from the right side.

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

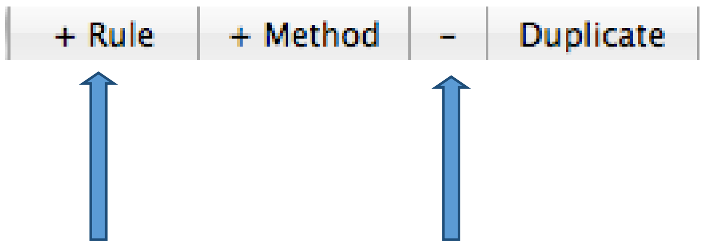


Press the UP ARROW key on the keyboard to get the up arrow indicator.

Have the students test their rule. Use these buttons in the top bar of the AgentCubes Online window to control the game.  Click on the green arrow to start the game. Have them press the up arrow to see if their frog moves up. Press the Red Square to stop the game, and the reset button  to reset the worksheet. When the student presses the up arrow, the frog should move up.

Show students how to add rules by clicking the **+Rule** button below the Frog Behavior. Also show them how to DUPLICATE rules using the DUPLICATE button.

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



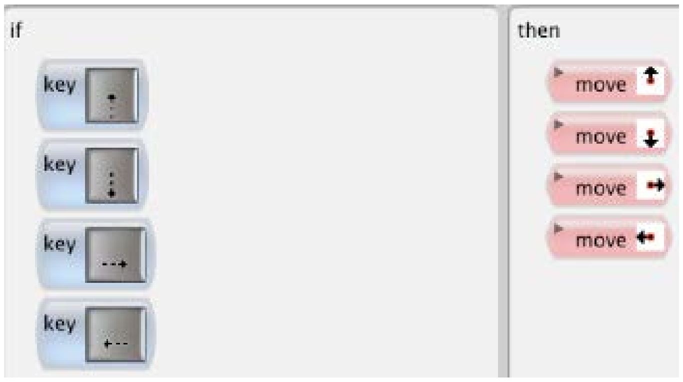
New Rule

Delete a rule

Select a rule by clicking in the center of it. Click on DUPLICATE to copy the rule. Click on the minus sign to delete a selected rule.

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. At this point, it wont take too long if they accidentally forget to save and need to re-create the entire workspace. They won't do it again. That being said, reminders to save are good too!

COMMON MISTAKE:



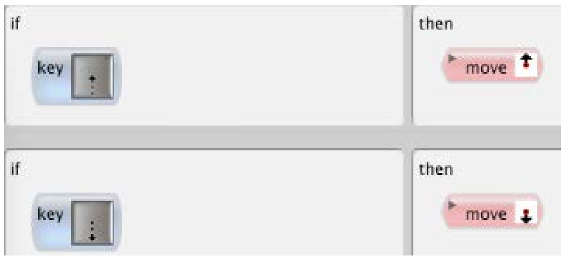
In this case, the agent is being 'told'…if I click the up, down, right and left arrow, then you move up, down, right and left.

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 down arrow, you move down.



They must each be their own, <u>separate</u> 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?

<u>Resist the urge</u> to find their mistake or take over their computer and fix their program!

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

Teacher Instructions:

Part 4b – 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?

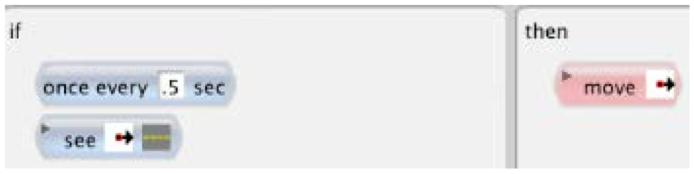
Common Misconceptions:
Listen for students suggesting that the **trucks** somehow make a **loop**…that they go from the tunnel on the right 'back around' to the tunnel on the left.

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 left 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 independently before providing assistance!

This is a great time for pair or small group work

Truck:



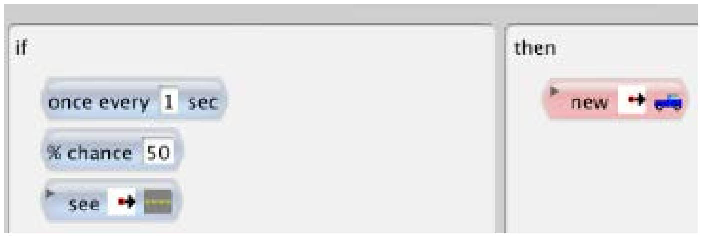
Explanation: We want our Truck to move to the right every so often, as long as there is street to the right. Drag the "See" condition and click on its depiction parameter to define that the Truck needs to see a Street and its direction parameter (arrow) to indicate that it needs to see the Street to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

Click on the Left Tunnel Agent and create a rule that looks like the following:



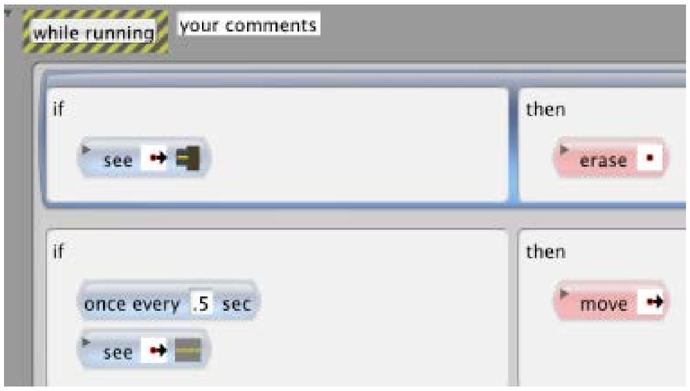
Explanation:

We want the Tunnel to create a Truck to its right, but only if there is street there. Otherwise, if there is, say another truck, 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 1 second." It also says that "this behavior has a 50% chance of happening." So every 1 second there's a 50% chance that the tunnel will create a truck agent on the street to the right. These conditions create a random pattern of trucks on the road so that it is a challenge for the player to get the Frog across the road without it getting squashed by a truck. The students can adjust these numbers to make the game more challenging.

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 +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. Run the game by clicking on the Green triangle in the AgentCubes Online tool bar.

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.

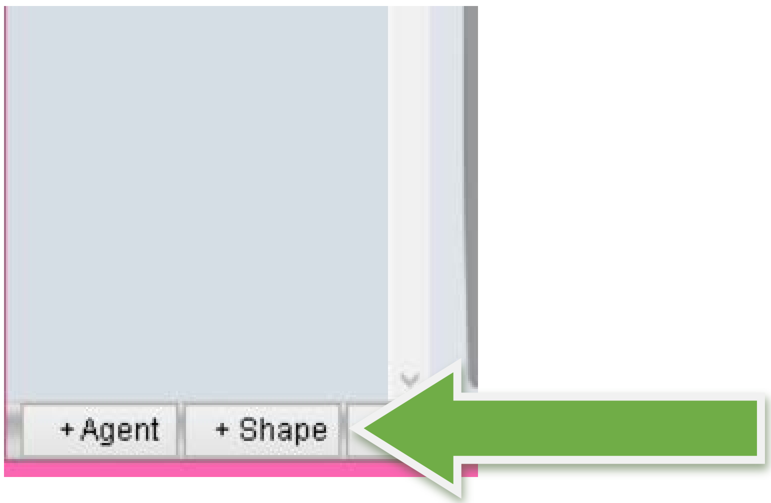
What happens? What's wrong?

Teacher Instructions:

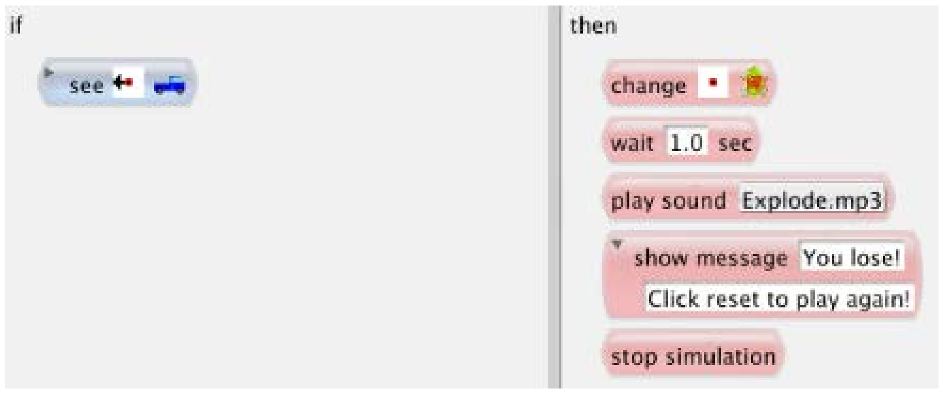
Part 4c – Truck/Frog Collision

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 shape that represents the squished frog to use when the 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. Select the Frog agent by clicking on it in the list of Agents. Below the list of agents, click on the +Shape button and name the new shape "Squished frog." Then, click on the picture next to the new shape name and use the drawing tools to edit the new shape so it looks like 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 change the Frog's appearance to look like the squished frog.
AND we wait 1.0 second (to give the program time to change the frog shape)
AND we play the "explode" sound
AND we show a message that explains that the player has lost the game
AND we stop the game

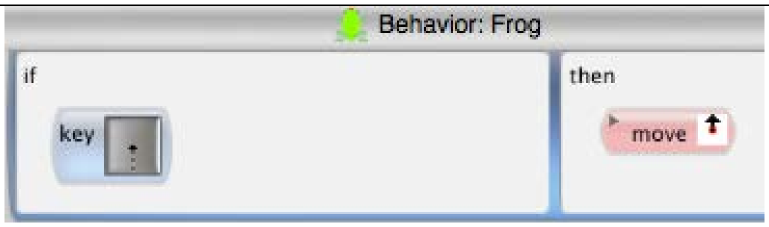
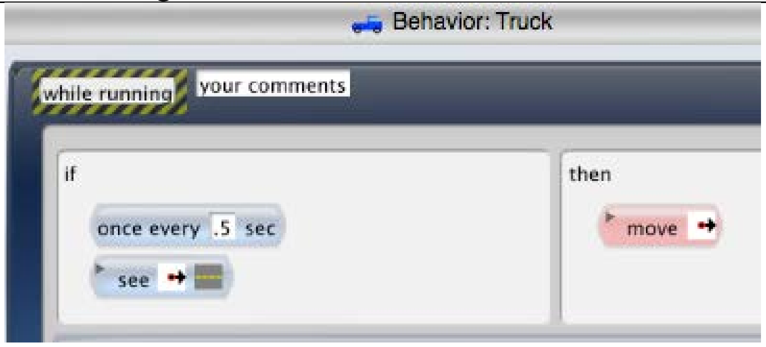
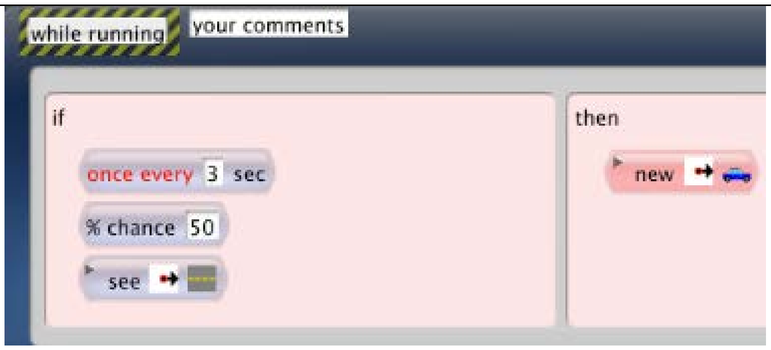
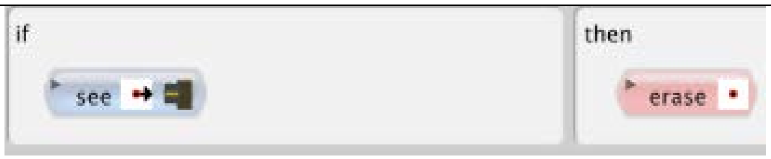
Reminder: Student Handout 4 details all of the behaviors for the agents. Consider whether your students need this support. Many students will be able to handle this programming without support once they have thought through the actions. ***(Found on page 10 in the student packet)***

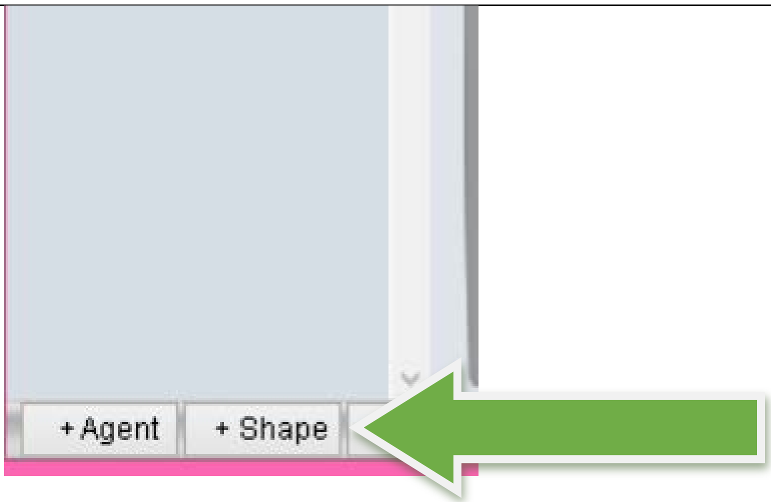
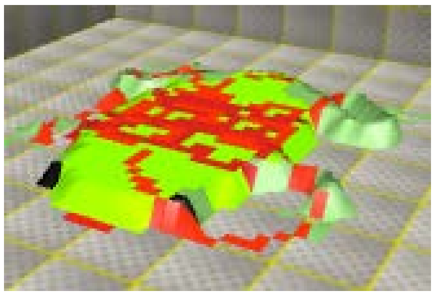
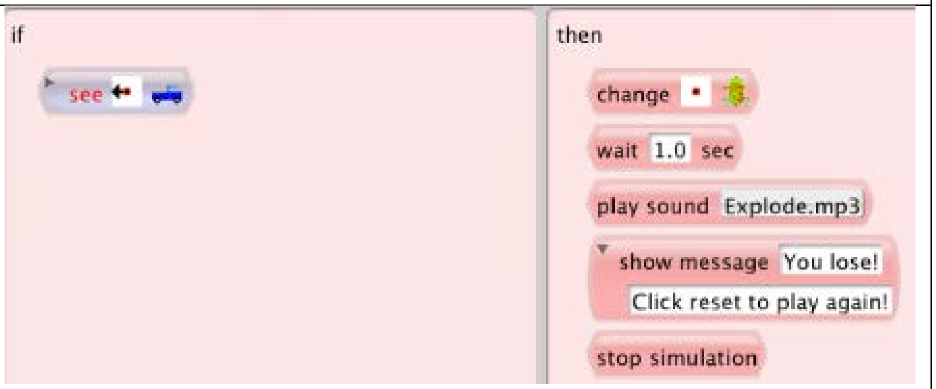
SCALABLE GAME DESIGN

3D-Frogger

Student Handout 4: Creating Agent Behaviors

Click on the agent to add behaviors to that agent

Step 1	**Open the Frog's behavior by clicking on the Frog in the list of agents or in the World.**	
Step 2:	**Cursor Control for Frog**	 **This rule makes the Frog move UP when you type the UP arrow. Use the +Rule button at the bottom of the window to add 3 more rules so that the Frog moves in all 4 directions.**
Step 3:	**Click on the Truck agent.** **Add a move right rule.** **Use the <u>once every</u> condition to slow the car down.**	 Behavior: Truck while running your comments if once every .5 sec see then move
Step 4:	**Add a rule to the left tunnel behavior to generate cars.** **Use the <u>once every</u> and <u>% chance</u> conditions to control how often they appear.**	 while running your comments if once every 3 sec % chance 50 see then new Change the values to make the game harder or easier!
Step 5:	**Add a rule to the truck so it erases itself when it sees a right tunnel.** **The tunnel <u>absorbs</u> a car.**	 IF I see the tunnel to my right…..THEN…erase me

Step 6:	**Create a 'squished frog'** **Click on the Frog Agent. Then click on the +Shape button below the list of Agents.** **Give the new shape a name. Double-click on the image next to the name.** **Use the drawing tools to make your Frog look squashed.**	 
Step 7:	**Play a sound and erase the frog when it collides with the truck.**	if see ← then change wait 1.0 sec play sound Explode.mp3 show message You lose! Click reset to play again! stop simulation **Note: the SHOW MESSAGE command has two lines of text – use the small arrow to see both lines!**

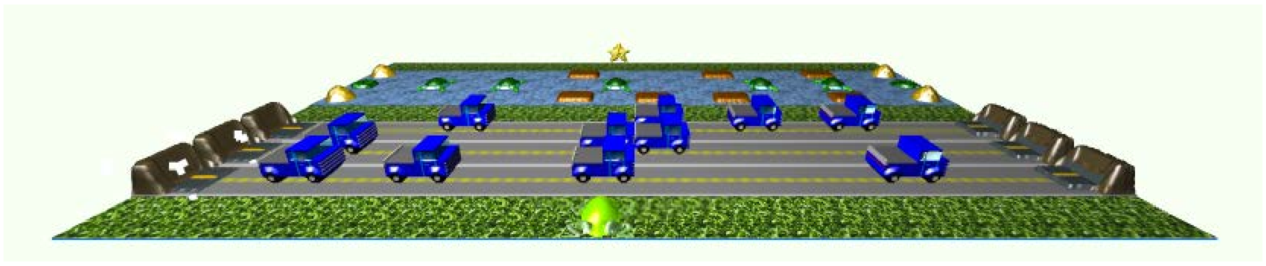
You are ready to move on once the following items work correctly…

- 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.



New Agents

- River
- Logs -- We will have these move from left to right.
- Log-Maker (The islands to the left of the river GENERATE (create) logs and the islands 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 lily pads GENERATE and ABSORB turtles)
- Grotto/Goal (the star at the top) so the Frog can beat the game

3D-Frogger (Continued)

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 (Island) and Turtle Maker (Lily pad): like the tunnels**
- **Grotto/Goal (flag): new**

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.

Added Rules of the game:

Have a class discussion or small group work to determine these rules… include discussions on how students might program these aspects, <u>but do not give the students all of the necessary code</u>. Also we encourage you to use the agent design worksheet.

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 (Island)

- 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 (Lily pad)

- 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/goal, the player wins!

Provide students with Student Handout 5 *(Found on page 13-14 in the student packet)*

Since this activity requires students to APPLY what they've already learned to a new (but similar) situation, allow students to do this on their own with guidance but not direction from the teacher. They should be encouraged to talk through problems with a neighbor.

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

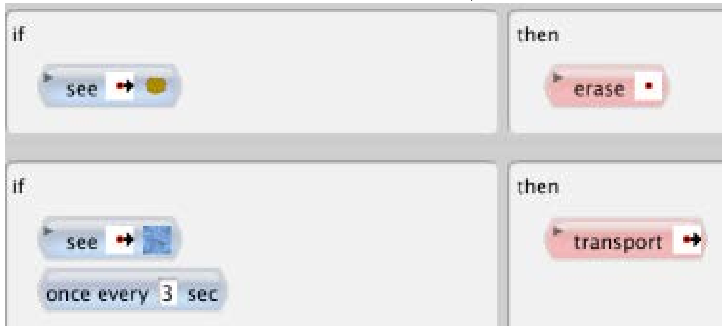
Agent: Frog:We must update the Frog Agent
- Jump on top of and move with the Logs and the Turtles
- The player loses if the frog falls in the water (the Frog Drowns)

Agent: Grotto/Goal
- If the Frog gets to the grotto, the player wins!

Step 1: Create missing agents (river, log, island, turtle, lily pad, grotto/goal) and add them to the worksheet.

Step 2: Program the island 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. See the log rules below. 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?

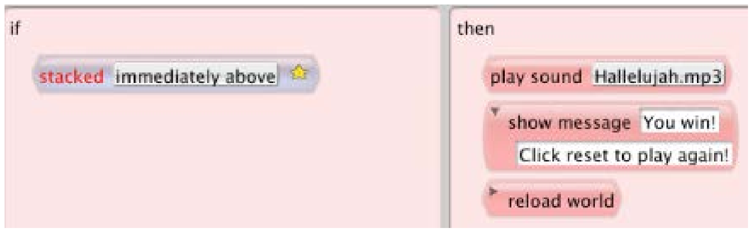
Step 5: Program the lily pad to generate turtles. Program turtles to disappear when they reach the end of the water.

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 goal. Give the frog this behavior:



if
stacked immediately above

then
play sound Hallelujah.mp3
show message You win! Click reset to play again!
reload world

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 15 in the student packet)

A) The main computational thinking patterns we covered are:
   1) **<u>Cursor Control</u>**: intentionally moving an agent.
      a. Using keyboard keys to move an agent.
      b. Example is moving the frog.
   2) **<u>Generate</u>**: create new agents on the screen.
      a. Use the "New" action in AgentCubes Online.
      b. Examples are generating new trucks, turtles, logs in Frogger.
   3) **<u>Absorb</u>**: deleting agents on the screen.
      a. Use the "Erase" action in AgentCubes Online.
      b. Examples are erasing the trucks, turtles, and logs on the other side of the screen.
   4) **<u>Transport</u>**: transporting an agent along with another, as if one agent is carrying the other.
      a. Use the "Transport" action in AgentCubes Online.
      b. Examples are transporting the frog on the logs and turtles.
   5) **<u>Collision</u>**: 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 AgentCubes Online are:
   1) Creating projects, worlds, and agents.
   2) Changing shapes for different circumstances, such as the 'squished frog' shape.
   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 16 in the student packet)

Challenge 1.0: Alligators

Alligators

Add alligators to the river.

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.

Tip:
Make sure the alligators move faster than the logs and that the alligators can climb up onto the logs.

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

3D-Frogger

Student Handout:

(Found on page 17 in the student packet)

Challenge 2.0: Prevent Cheating

No Cheats

Prevent anyone from cheating in order to win the game!

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 it rides all the way to the end of the water
The frog should not be able to walk on tunnels or islands or lily pads

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 18 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 left
- Frogs are squished if hit by the cars
- Squished frog means the end of the game

Student Handout:

(Found on page 19 in the student packet)

Challenge 4.0: Inquiry Element

Inquiry Element

?

Make up a fun new element for your game

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:
Make up a new fun element to your game, assign new behaviors, create new agents, and/or make new depictions. The only requirements are that it is new and that it is fun.

Student Handout:

(Found on page 20 in the student packet)

Challenge 5.0: Re-design the road

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.

Road Design

Make the road look more realistic

Design Challenge: There's something funny looking about your roads. Figure out what it is and fix it.

New Agents:
Create a new road shape for a top and bottom of a two (or more) lane highway. Potentially create new tunnels and cars to go in both directions (if you didn't already do challenge 3.0)

Update Worksheet

- Put new roads down so cars and trucks are only in one lane
- Save the worksheet

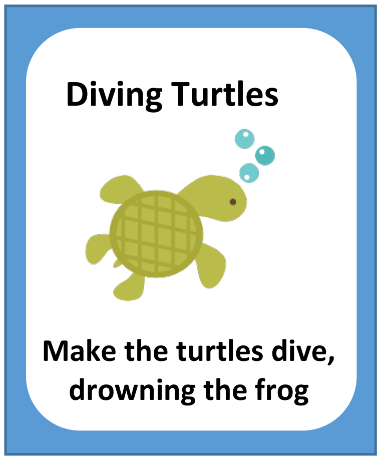
Student Handout:

(Found on page 21 in the student packet)

Challenge 6.0: Diving Turtles

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: Turtles don't always swim at the surface. Sometimes they dive. This would cause the frog riding on them to fall in the water.

New Agents:
Underwater turtle shape

Update behaviors

- Make the turtle sometimes dive, changing it to an underwater turtle
- Make underwater turtles re-surface
- Make the frog drown if it is on an underwater turtle

ISTE Standards[3] 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.

[3] 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 derivate 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

3D-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), AgentCubes Online (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.