# Frogger

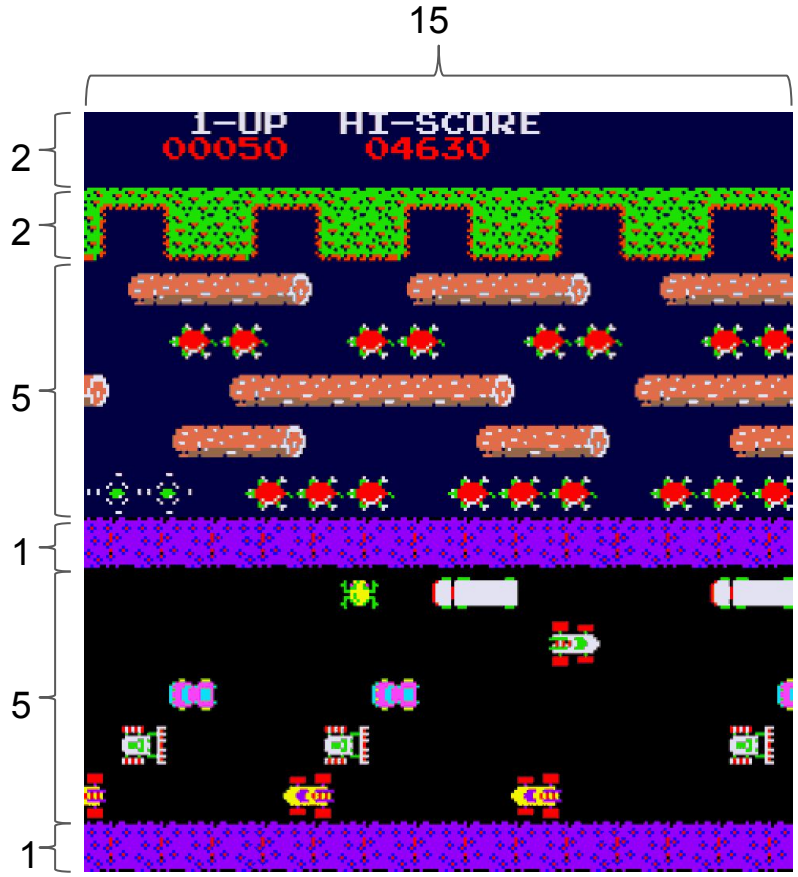
By: Martin Gutierrez

**ITCS 4230/5230 - Introduction to Game Design & Development**

**UNC Charlotte**

# Set up - The room

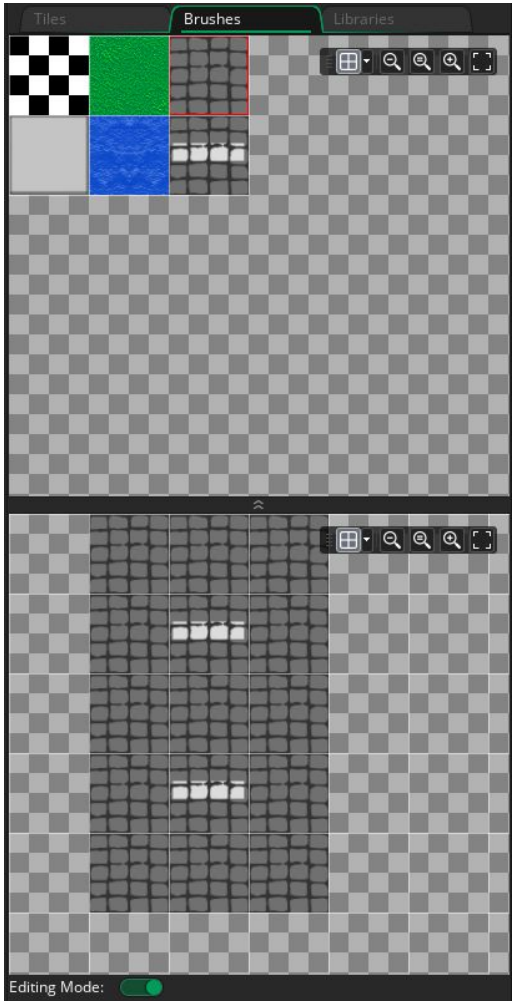
- Download the Frogger Partial file
  - Notice that the file includes all sprites
  - Notice that the file has 1 object called obj_Frog
    - Look at the events in the obj_Frog. Make sure you understand what these events are trying to accomplish.
- Create a new tileset, name it ts_Level1
  - Apply spr_Level1Tiles to ts_Level1
  - When you set the sprite, in the tile set properties, set the tile width and tile height to 64.
  - Create a new tile layer and set ts_Level1 to it
- Tile the room according to the original frogger
  - [Here is a video of original Frogger gameplay]



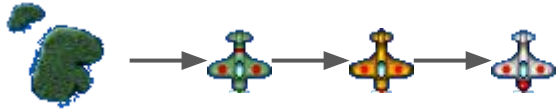
# Side Note - Custom Brushes!

Go to the room editor by double click on your room

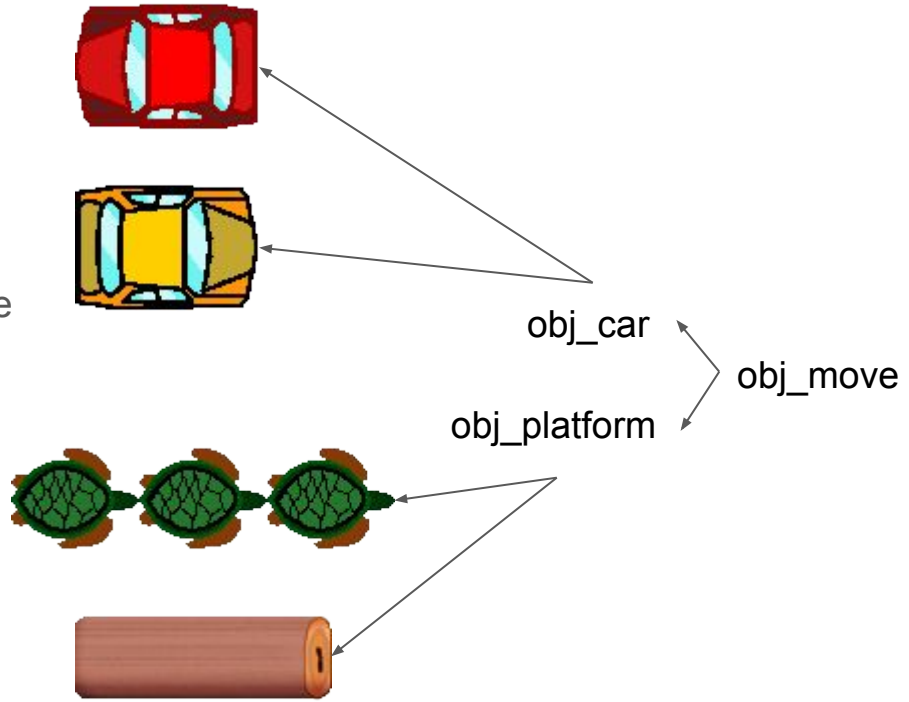
- Make sure you are in the tile layer!
- You will see three tabs. Click on the brushes tab.
- Notice the two windows. The top one is your tiles window. The bottom is the custom brush window.
- To create your own custom brushes turn editing mode on by clicking on the slide at the bottom left. Gray = off, Green = on.
- Set tiles in the custom brush window. Once done turn editing mode off.
- Click on your newly created brush in the custom brushes window.
- Set it inside of the room! You have created your own custom brush! This will save you time when setting street tiles.



# A quick note on inheritance



- Recall that in previous workshops, we would create one object with all of its functionality, then use that object to be the parent of other objects.
- In workshop 1, inheriting the plane's behavior from the island made sense. The island did not have any functionality that would interfere with the plane's behavior. But what if it did?
- In this workshop, all cars and platforms will move in the same way, but interact with the player differently.
- We will still use inheritance, but we will need to use a different hierarchy than the straight line we've been using.

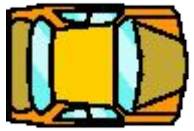


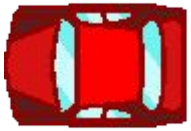
# Creating the parent object

- Create a new object, named obj_move.
  - Built in variables can be set directly in variable definitions. If we create a variable definition and set its name to hspeed, we can edit hspeed as if it were any other variable definition
- On Step event
  - When the mover exits the screen, jump it to the other side
    - Hint: Think back to how we checked if an island moved off the screen in workshop 1.
    - Hint: Use the sign of hspeed to determine which side of the screen that the mover will be concerned with crossing.
- To test that this is working properly, you can create a spr_debug to allow it to show up on the screen.
  - Go ahead and set hspeed to positive and negative values to make sure it interacts with both sides of the screen

# Adding the cars - obj_car

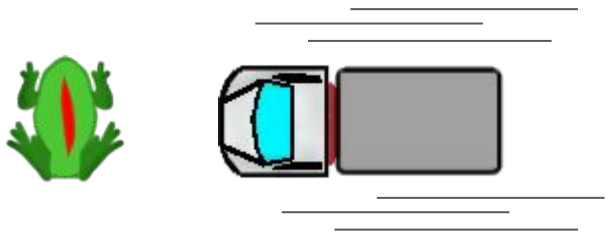

- Create six new objects. Name one obj_car and the rest obj_car1 through obj_car5
  - Set obj_car as a child of obj_move, and the remaining objects as a child of obj_car
  - Set obj_car1 through obj_car5 with their respective sprites
- Set hspeed to the following values:
  - obj_car1: -10
  - obj_car2: 10
  - obj_car3: -12
  - obj_car4: 25
  - obj_car5: -6
- Drag and drop the cars into the room
  - Check if they behave as they should. If not, test that obj_move is working as intended, and test that each object is inheriting correctly.





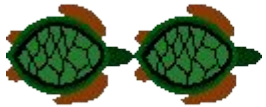
# Adding the cars - Collision with obj_car

- Go back to obj_frog. Here, create a Collision event with obj_car. In this event, have the instance of obj_frog destroy itself.



# Adding the platforms - obj_platform

- Create 4 new objects named obj_platform, obj_turtles1, obj_turtles2, obj_log
  - Set obj_platform as a child of obj_move, and set each platform as a child of obj_platform
  - Set obj_turtles1, obj_turtles2 and obj_log to each corresponding sprite
- Set hspeed to the following values:
  - obj_turtles1: 10
  - obj_turtles2: 5
  - obj_log: -5

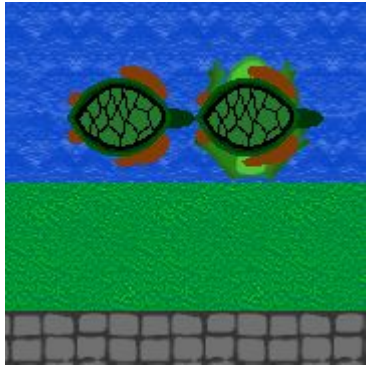


- Go back to obj_frog and go to the Step event
  - If there is an instance at obj_frog's x and y position, set obj_frog's hspeed equal to the hspeed of the platform
    - Hint: instance_place() returns a reference to the other instance. Check the link for how you might access the other's hspeed...

- Drag and drop the turtles into the water
  - Try riding on top of the turtles to see if they are behaving as they should

# Problems?

- Notice what happens when the frog jumps on top of the turtles. The frog keeps moving to the right even when it hops off.
- Also, the frog is underneath the turtle
- In order to fix the first problem: Recall what the frog does. The frog only checks to see if it is on a platform.
  - Hint: Try the *else* function to set the frog's speed back to 0 when it hops off.
- In order to fix the second problem: Create a new instance layer and name it "safe". Create one more instance layer and name it danger.
  - Delete the cars and the turtles from the room.
  - Click on the danger layer and drag the cars into the room.
  - Click on the safe layer and drag the turtles into the room.
- Remember to keep objects in the correct layers.



# Adding the platforms - logs

- Refer to the screenshot, there are 3 different logs in the original game. We only have one obj_log object. This means that we're going to have to edit the variable definitions inside of the room to give them different speeds.
    - For simplicity's sake, we will be referring to each log by the row that it is on. The logs on the bottommost row will be called log1, middle row will be log2 and topmost row will be log3.
- Log2 is the same as log1 except that it moves at a faster speed and is longer.
    - Inside the room, double click on an instance of obj_log. Set the Scale X to 2, then click "variables." Set hspeed to -10 here. This will only change the hspeed of this specific instance.
- Log3 is the same as log1 except that it moves at a faster speed.
    - Again, use variables to set the hspeed of logs in the topmost row to -10.

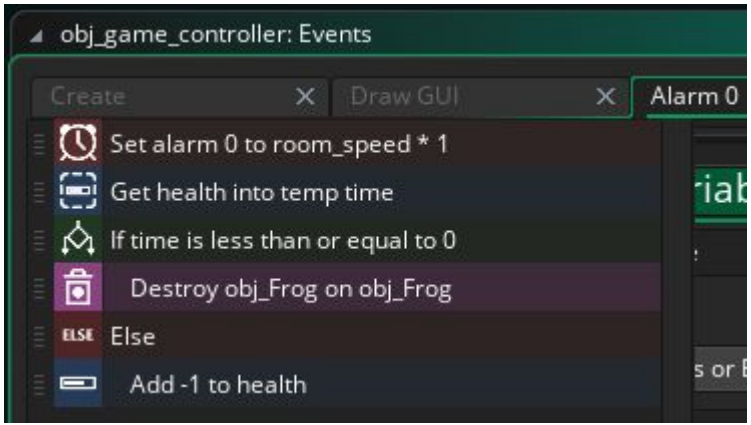
# Rearrange

- Remember the picture of the original frogger. Notice the amount of cars, turtles and logs.
- Rearrange your room to look like this.
  - For simplicity's sake:
    - Car1: 4
    - Car2: 3
    - Car3: 3
    - Car4: 2
    - Car5: 3
    - Turtles1: 3
    - Turtles2: 3
    - Log1: 3
    - Log2: 1
    - Log3: 3

- Refer to the screenshot to ensure everything is lined up.

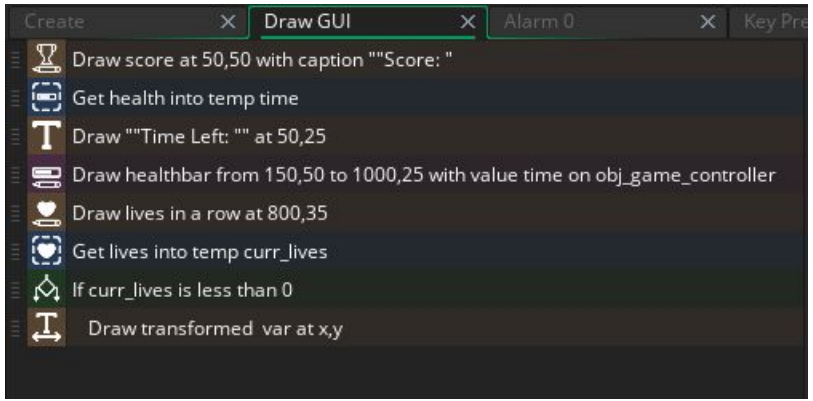
# Adding the Game Controller

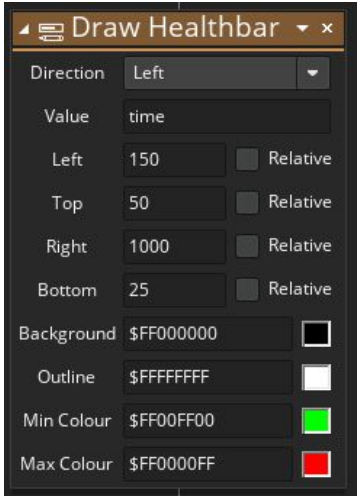
- The game controller will dictate our progress
- Create a new object called obj_game_controller
  - On create set score to 0, set lives to 3, set health to 30. Also set an alarm to 1 second. You can do this with the function: room_speed * 1
- On alarm event
  - Reset the alarm to one second
  - Get health into temp time
  - If time <= 0, destroy the frog
  - Else add health -1

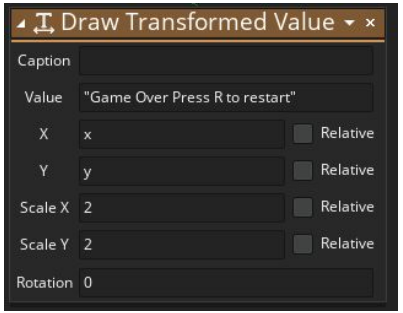


# Adding the Game Controller (cont'd)

- On Draw GUI event




- On key press - R
  - Restart Room
- Drag and drop the game controller into the room. Check to see if it works.(Time is going down, lives are showing, **R** restarts room)

# Adding the Danger zone

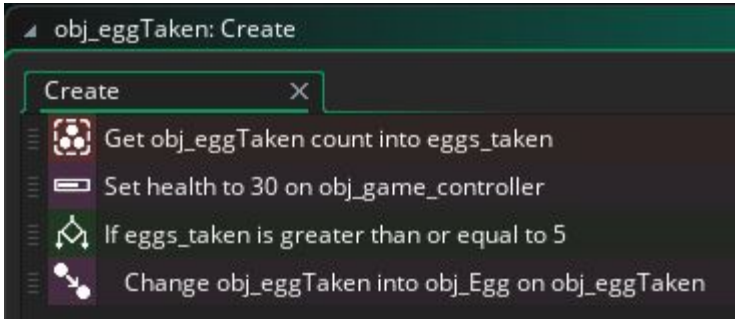
- The danger zone will be what we use to destroy the frog if it jumps into the water.
- Create a new object called obj_dangerZone
  - Set sprite to spr_danger, this is a blank sprite, we will be scaling it when we drop it into the room.
- Go to obj_frog and create a new Collision event with obj_dangerZone
  - If there is no obj_platform making contact with the frog, destroy the frog
    - Hint: Use If Object At or instance_place()

# Adding the Danger zone (cont’d)

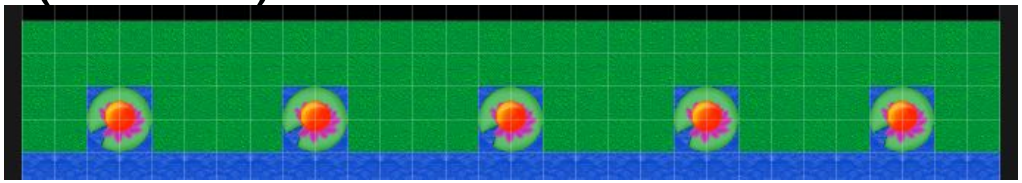
- Drag and drop obj_dangerZone into the room.
  - The sprite will be invisible, so make sure that you remember where it was
  - Once you click on it you will see the familiar blue outline, scale this outline to the dimensions shown in the picture.
- Run the game
  - Check to see if you die on water
  - Notice that you can still jump on turtles and logs

# Adding the Salmon Egg


- The salmon egg is our goal. This will give us points.
  - Create a new object called obj_Egg
    - Set sprite to spr_egg
    - Set as a child of obj_platform and set hspeed to 0
  - Create another new object called obj_eggTaken
    - Set sprite to spr_eggTaken
- On obj_Egg
  - Collision with frog, jump obj_Frog to its start position, add 25 to the score relative to obj_game_controller, change the instance to obj_eggTaken
- On obj_eggTaken
  - On create:



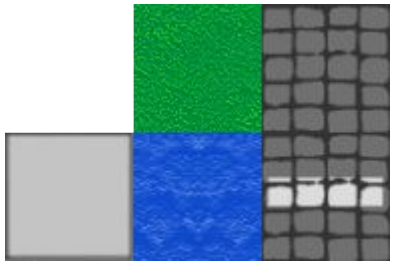
# Adding the Salmon Egg (cont'd)



- Drop obj_egg into the room
  - Touch the egg with the frog
  - The egg should change into obj_eggTaken and the frog should jump back to the start.
  - Notice that the timer also resets and the score is added 25.
- Egg Placement
  - Take this time to check if the eggs are properly working. Place the eggs into their proper spots located in the 5 crooks at the end of the map.

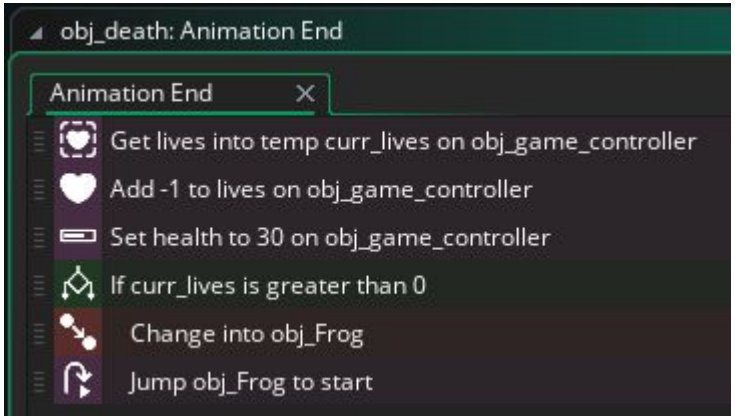
# What is left?

- Notice that we have lives, but do not respawn. We will add a death object that will allow us to do this.
- Before moving on please check to see if all mechanics are working as expected.
- Also take note of why the functions we used work as they do.



# The Home Stretch - obj_death

- Create a new object called obj_death and set the death sprite to it
  - We will use this object to respawn us if we have lives left. It will also handle when we lose lives.
- On animation end:




- Once this is done, we need to go back and change all of the *Destroy Frog* functions into *Change Instance functions*.
  - The Cars, the danger zone, the game controller.

# The Home Stretch – Adding Music/Sound Effect

- Add your own flare to the game by adding:
  - Background music
  - Sound effects

**This is your chance to be creative!**

# The Home Stretch - Minor Fixes

- Run the game
- When you run out of lives the obj_game_controller tell the player to press **R** to restart.
    - The position of this text is based off of the position of the obj_game_Controller. If the text is in a bad position either move obj_game_Controller into a better position or directly change the x and y values in the code.
- Refer to the screenshot to get placement of any of the instances, especially the cars, logs and turtles.

# The End~

Enjoy!

Tutorial by: Martin Gutierrez