



# GAME PROGRAMMING & DESIGN – LAB 2

## Animal Olympics - another simple SCRATCH game

---

### **BACKGROUND**

#### **I. Introduction:**

In our last lecture we talked about the evolution of high-level programming languages and discussed popular programming paradigms. We discussed in some detail three of the most popular paradigms:

1. Imperative (the intelligent list)
  - a. Sequence
  - b. Selection
  - c. Repetition (Iteration)
2. Procedural (making calls)
3. Object Oriented ( a program as a group of interacting objects)

Today you will create a simple game in Scratch using the concepts behind these paradigms.

#### **II. Ludology & Funativity:**

As you create the game you should make a determination on whether or not YOU think the game is fun. If you think it IS fun, why? If you think it IS NOT fun, why? We talked about abstract (psychological) things that make a game fun and concrete (physical) things that make a game fun. As you create the game, ask yourself how the following concepts are demonstrated (or NOT demonstrated):

Abstract (things that make a game fun):

- Spatial Relationships
- Pattern Recognition
- Social Interaction

Concrete (things that make a game fun):

- Goals (Numerous, Clear, Accomplishable)
- Rewards and Punishments.
- Choice (or the illusion of choice).

### **PLANNING THE GAME**

#### **1. Description of Game:**

*(Never start a game by sitting down and coding. Every large programming project needs to be planned out, in as much detail as possible BEFORE you start coding. The more time, effort and detail you put into your planning stage, the faster the coding of the game will actually go.)*



The game will be called **Animal Olympics**. In the game a cat (the player) will attempt to win a variety of track and field competitions, in order to be declared Olympic champion. The player will control the cat using the keyboard.

In the first iteration of the game (what this tutorial covers), we will create the 100 meter dash part of the game. In this part of the game the player will run a race. To make the cat run the player must alternately push the left and right arrow keys. If the cat reaches the finish line first the cat wins.

## **2. Programming Outline**

Plan, plan, plan, before you write a single line of code! A programming outline will help you see problems, and help prevent you from making mistakes

<u>Object</u> (Name, Description)	<u>Properties</u> (What are the facts about this object? What does the object look like? How many images will you need for it? Where does it start? What are its states (alive, dead, etc.)	<u>Functions</u> (What does this object do? Can it move? Can it change costumes? Can it interact with other objects? Can it interact with the player?)
<b>Stage</b> (the stage itself, which will have a title and a gameplay screen).	Stage will have two backgrounds. Stage will have two variables: canGo catWin both of these will be set to 0 at start.	Stage will switch backgrounds, play a sound, set canGo to 1 and broadcast a <b>startgame</b> message to start the actual game.
<b>Cat</b> (the player)	Will have two costumes so that it can appear to run. Will have one variable: lastKeyWasleft which will be use to control player input.	The cat will move and change costume when the player alternates between pushing the left and right arrow.
<b>Bat</b> (the opponent AI)	Will have two costumes so that it can appear to fly.	Will automatically change costume and move left to try and beat the player.
<b>Finish Line</b> (will determine who wins, based on who touches it first).	Has only one costume.	Will detect whether the bat or the cat touches it first, it will then set the canGo variable to 0, stopping the game, and set the mouseWin variable to 1 if the cat was the first to touch it.

## **CREATING THE GAME**

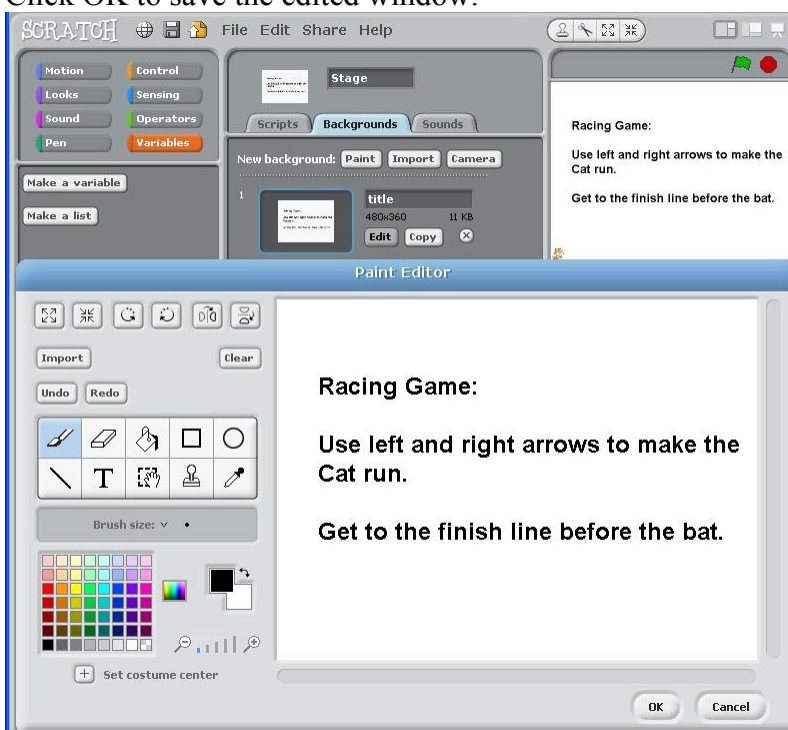
### **1. Creating the Objects:**

By looking at the programming outline, you should be able to see that we need four objects. In SCRATCH these objects are called Sprites.

1. Start a new game. This will give you two of the sprites you will need automatically: the Stage and the Cat.
2. Create the two backgrounds for the Stage sprite:



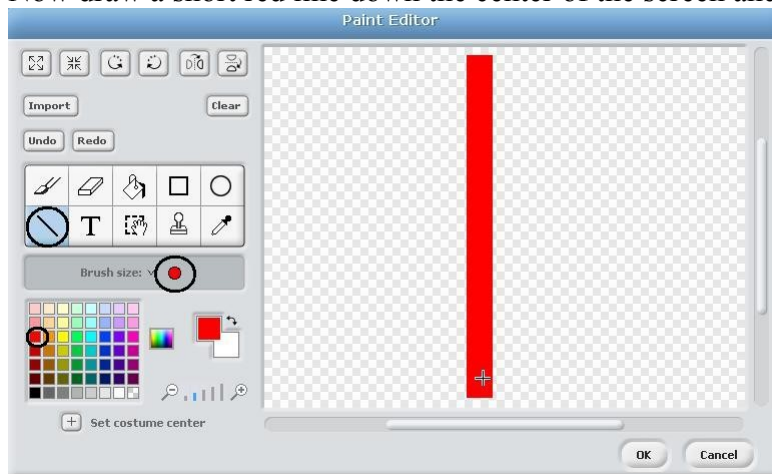
- a. Select the stage, by clicking on the **Stage** icon in the Sprite window (lower right of screen).
- b. Click on the Backgrounds tab in the center of the SCRATCH Window.
- c. Rename the background that is already there (background1) to title.
- d. Click on the edit button underneath the **title** background.
- e. In the Paint Editor window that appears, select the **Text Tool** button in the left panel.
- f. Add the following text to the title window:  
Racing Game: Use left and right arrows to make the Cat run. Get to the finish line before the bat.
- g. Click OK to save the edited window.



- h.
  - i. Add a second background image by clicking on the Import button. I used the xy-grid. You can add whatever you want, just change the name to **playscreen** afterward.
  - j. Now add a sound to the Stage.  
Click on the sounds tab, and then click on Import.  
Click on the **Music Loops** folder and choose the **GuitarChords1** sound file. Then click OK.
3. Create the Bat sprite,
- a. Click on the **Choose New Sprite From File** button (the middle button) in the sprite area.
  - b. In the costumes folder click on the Animals folder, and then click on bat1-a, and click OK.
  - c. With the new bat sprite selected, click on the Costumes tab in the middle of the screen, you should see one costume.
  - d. Add a second costume by clicking on the Import tab, choose the bat1-b costume.



- e.
4. Create the Finish Line sprite.
    - a. Click on the **Paint New Sprite** button.
    - b. In the Paint Editor window that appears, choose the **Line Tool**, a medium size brush, and the color red.
    - c. Now draw a short red line down the center of the screen and click OK.



5. You should now have 4 sprites (including the stage) in the sprite area of the application.



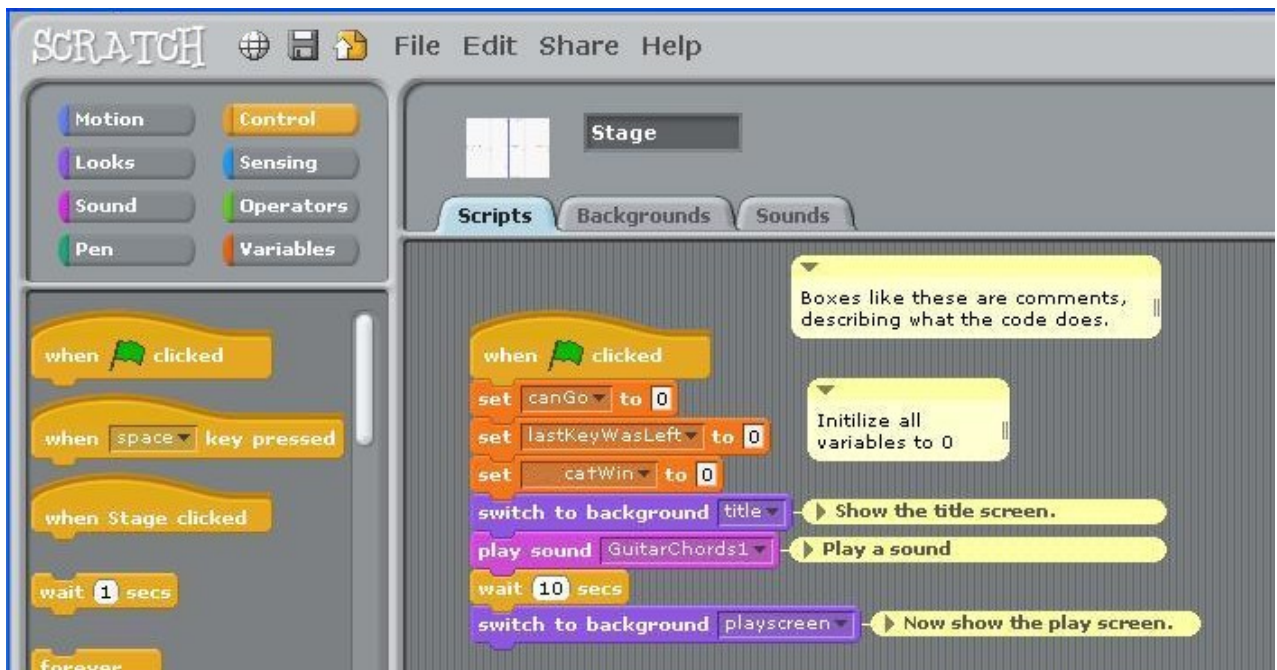
6. You now need to add three variables to the game.



7. Congratulations, you have created the 4 sprites (objects) that you need!

## **2. Sequence**

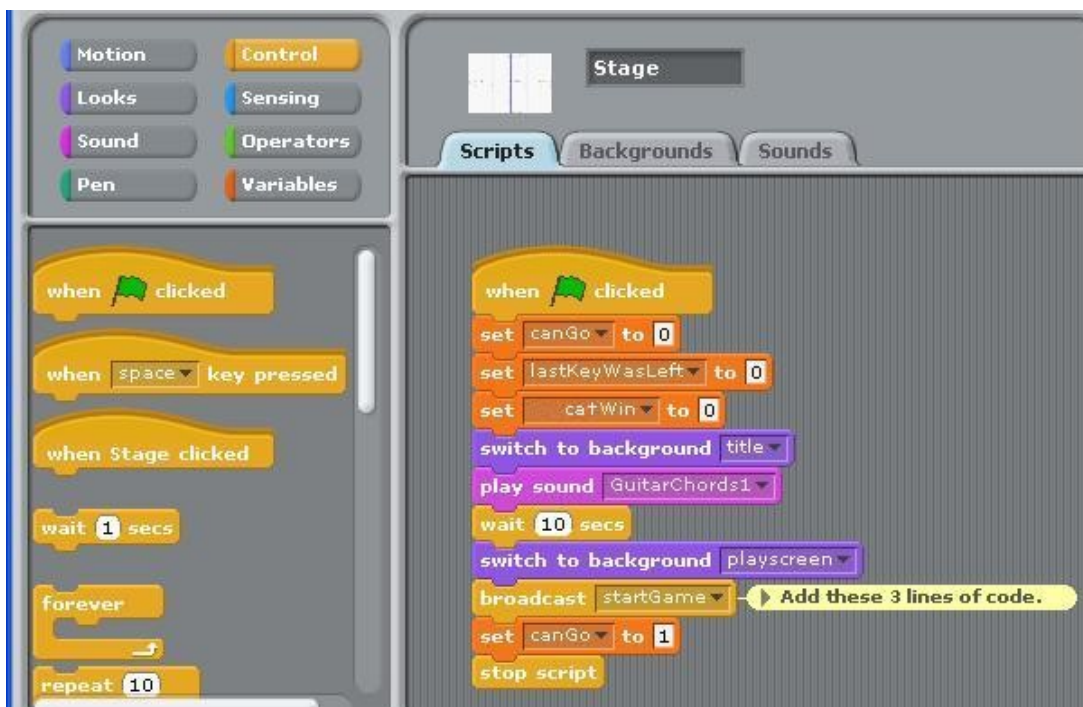
The code in SCRATCH is put together using blocks from the library and is assembled in the Scripts area for each Sprite. The scripts that are created are read from top down once they are started. We need a script for the stage, that will initialize all of the variables to 0 (by convention, 0 means NO and 1 means YES), display the title screen, play a sound, wait 10 seconds, and then switch to the playscreen. Click on the stage icon in the sprites area, and then click on the Scripts tab in the center window. Then drag and drop the code blocks from the library on the left of the screen, until your script window looks like this (YOU DO NOT NEED TO ADD THE COMMENT BOXES):



Once your script looks like the example above, click on the green flag in the top right of the screen to test it.

## 2. Procedures (messages)

We need a way for different objects to communicate with each other in the game. As an example, after the Stage sprite has switched to its playscreen background we would like it to send out a signal to all of the other sprites that the game can now start. Add the following code to your script in the Scripts window of the Stage sprite.





Once your script looks like the example above, click on the green flag in the top right of the screen to test it.

## 2. Selection and Repetition

Select the bat sprite. He needs to be put in his starting place and hidden when the green flag is clicked. Then he needs to be appear, and start flying left when the game actually starts. Make your code look like this:

The screenshot shows the Scratch IDE interface for a bat sprite named 'Sprite2'. The sprite's current position is x: 190, y: -50, and its direction is 90. The 'Scripts' tab is selected, showing two scripts:

- Initialization Script:** This script is triggered when the green flag is clicked. It contains the following blocks:
  - when green flag clicked
  - set size to 30% (comment: Make the bat smaller.)
  - point in direction 90 (comment: Make the bat point to the right.)
  - go to x: -220 y: -50 (comment: Put the bat in the lower left of the screen.)
  - hide (comment: Hide the bat.)
  - stop script
- Game Script:** This script is triggered when the sprite receives the 'startGame' message. It contains the following blocks:
  - when I receive startGame
  - show
  - forever if loop with condition 'canGo = 1' (comment: IF (selection) canGo is 1 (true) then repeat the following lines of code). The loop contains:
    - next costume
    - wait 0.2 secs
    - move 5 steps

Once your script looks like the example above, click on the green flag in the top right of the screen to test it.

## 2. User Interaction

Select the cat sprite. He needs to be put in his starting place and hidden when the green flag is clicked. Then he needs to appear and respond to the users input (pushing the left and right keys) when the game actually starts. Make your code look like this:

The screenshot shows the Scratch code editor for a sprite named "Sprite1" (a cat). The top panel shows the sprite's position: x: 195, y: -140, direction: 90. Below are three tabs: Scripts, Costumes, and Sounds. The Scripts tab is active, showing three script blocks with explanatory text boxes.

**Initialization Script:** This is the initialization script. It will get the bat back to its starting position.

```
when green flag clicked
  set size to 45 %
  point in direction 90
  go to x: -220 y: -140
  hide
  stop script
```

**Start Game Script:** This script makes the cat appear, and has him/her tell the player what to do.

```
when I receive startGame
  show
  say Let's Go! for 2 secs
  say Hit the left and right arrow keys! for 6 secs
  stop script
```

**Game Movement Scripts:** These are the game scripts and allow the user to move the cat by hitting the left and right arrows. Notice how the if statement has two conditions that must both be true in order for the code inside to work.

```
when left arrow key pressed
  if canGo = 1 and lastKeyWasLeft = 0
    next costume
    move 5 steps
    set lastKeyWasLeft to 1
  stop script
```

```
when right arrow key pressed
  if canGo = 1 and lastKeyWasLeft = 1
    next costume
    move 5 steps
    set lastKeyWasLeft to 0
  stop script
```

Once your script looks like the example above, click on the green flag in the top right of the screen to test it.

## 2. Collision Detection

The finish line needs to know who hits it, so that it can stop the game and determine whether or not the cat won. Make your code look like this:



### **Making the Game Better:**

What could be done to make this game better and more fun to play? Trying adding some more things to this game:

- Add a screen at the end that tells whether or not you won or lost.
  - (Hint: Use the gameOver broadcast and the catwin variable).
- Add a difficulty setting.
  - (Hint: Create a variable to specify how long the bat pauses between moves).



- Add another opponent.
  - Make the new opponent start slow, but get faster as the race progresses.
- Add a start button after the Title Screen.
- Add some more sounds (running, wings flapping, etc.)
- Add a second race (a second level that will start after a **secondlevel** broadcast is made).
  - Give the cat the ability to jump.
  - Add hurdles on the course that the cat has to get over.