Behind the Digital Screen
Assignment 3: Program a Game

Deadline: Monday, 3/14, 3:00pm (one hour before class begins)

Objectives:

- Demystify computer programming and obtain a conceptual foundation to build upon when learning other programming languages.
- Complete your first interactive computer program (a game) using the Scratch programming language and share it online.
- Demonstrate an understanding of fundamentals of computer programming that are useful in other languages. (These include: statements, operators, variables, arrays, control structures, events, random numbers, parallelism, algorithms, and commenting. Learn to effectively reuse and purpose existing source code.)

Lab Report:

We're changing it up! So that you don't forget your knowledge of HTML from Assignment 2, you will enter **your lab report** into an HTML] file that you create on your own, which must be turned in with the assignment. *Create this file with a text editor (e.g. Notepad++, TextWrangler) and name it "assignment3.html"* (minus the quotes). You will be asked questions during steps **A8, A26, B5, B15,** and **C** (your source list for the whole of Part C, then your final question)**. Also include your bonus work,** if any**.** Enter your responses to these questions in this HTML file and save it. You can format this HTML page any way you wish, as long as your answers are easily readable by the teaching staff when the file is opened with a web browser. For instance, in your HTML file you could include all your responses in separate paragraph sections within the <body> of your HTML document, like this:

```
<p>Question A10: (Your answer here.)</p>
<p>Question A31: (Your answer here.)</p>
(…)
```

Or embedded in a table, like this:

```
<table>
  <tr>
    <td>Question A10:</td>
  </tr>
  <tr>
    <td>(Your answer here.)</td>
  </tr>
  (…)
</table>
```

Or however else you'd like to mark up your text using proper HTML tags. Just be sure all text responses are contained within the single HTML file that is named assignment3.html and that they appear properly and readable when opened with a web browser. You do NOT need to create a CSS file, just a single simple HTML file containing your assignment notes/responses.

Files to turn in:
In this assignment you will be producing multiple files (**one HTML file, 12 Scratch files, and multiple image and/or audio files**). The Scratch files you create will be uploaded to the Scratch website (http://scratch.mit.edu) where they will be publically available for the world to see. To do this, you will need to **click the "share" button** while editing each project. You will also be uploading the files via Canvas as usual. These files will include:

- ☐ One **HTML** file (assignment3.html) containing **your lab report** (text responses) and links to each Scratch project (be sure you "shared" them).
    - Eight scratch projects you create in Part A
    - Three Scratch projects you create in Part B
    - One Scratch project you create in Part C
- ☐ The Scratch **code** for these "shared" projects. When you are done, you can obtain the code from the Scratch interface: click "File" | "Download to your computer" to save the files. They should have a .**sb2** extension. You should have one .sb2 file for each project listed in the previous bullet point.

<assignment>

Part A: Your First Program

In this section you'll be learning how to use the *graphical programming language* Scratch. Some of the steps below require you to write a text response. **Save these text responses** in a word processor, text editor, draft email, or anywhere else you can save text. They will be entered in the HTML file you create named assignmenet3.html, to be turned in with the rest of your files.

This assignment assumes you have **no prior use of the Scratch programming environment**, however, it assumes you have gone through the Scratch readings and the following Scratch learning materials in Part A to obtain a basic understanding of what Scratch is, how it works, and tips/tutorials for how to create and share Scratch projects.

Basics and Interface

1. Create an account on Scratch. Go to https://scratch.mit.edu/ and click "Join Scratch", then follow the prompts that it gives you.

2. Skim the Scratch Getting Started Guide in full (16 pages, mostly screenshots – should go very fast) located here: http://d.umn.edu/~sivelab/project/learningToCode/Getting-Started-Guide-Scratch2.pdf

3. Watch "Getting Started With Scratch" (2:24 -- so that you understand the editor, how to browse projects, creating a new project, "seeing inside") and browse the "Introductory Tutorials" videos located at https://scratch.mit.edu/help/videos/

4. There are 12 'Scratch Cards' that demonstrate some of the most common ways to activate and manipulate content when programming in Scratch. Look at all 12 cards. These will give you ideas for the Scratch projects you will create in this assignment. Scratch Cards are located here: https://scratch.mit.edu/info/cards/

5. Navigate to the main Scratch homepage and browse through the 'Featured Projects' and other Scratch projects by clicking on the links from the Scratch home page: http://scratch.mit.edu/. **Something to notice** when encountering a Scratch project on the Scratch website for the first time are the **"Instructions"** and **"Notes and Credits"** often listed in the column to the right of the project itself. For interactive projects, these should tell you how to begin (e.g. use arrow keys to move, etc.). Be sure you know where **Instructions** and **Notes and Credits** appear on a project page before moving on.

6. Click the tab labeled as your username. A section called **"My Stuff"** will appear. Your completed Scratch work will go here.

7. **Create your Lab Report HTML file** now using a text editor (e.g. Notepad++, TextWrangler) and name it assignment3.html. Be sure to include the proper HTML basic structure you learned in Assignment 2 (doctype declaration, <head>, <body>, etc.). You will enter all of your assignment notes in this file and later turn it in with your assignment email submission.

8. Now that you have seen many examples of what Scratch is capable of, on the Scratch website **locate three projects that you think are worthy of sharing** and save the URLs corresponding to each of the projects. The URL will resemble something like: https://scratch.mit.edu/projects/82443924/. These can be any three projects, just as long as you think they are somehow interesting, funny, educational, aesthetically beautiful, technically amazing, etc. At this point you have likely noticed that there is a lot of junk in the Scratch website, but also some amazing projects. Browse the "Explore" feature (you can also use the search bar at the top) and find three Scratch projects that are inspirational and worth sharing with the entire class. **Record these three links in your Assignment Notes HTML document.** Use an appropriate heading to accompany the links within your HTML file (e.g. <h3>Three Amazing Scratch Projects</h3>).

9. In addition to the resources on the Scratch tutorials, the Scratch users' community maintains several pages which contain a wealth of resources, explanations, Q & A forums, and other helpful information for how to get things accomplished when programming in Scratch. Browse the **help** page from the Scratch home page and the **discussion forums** pages.

PRO TIP: The user community for a programming language is a key source of help when you are stuck on a programming problem. Every programming language has a community of users that talk to each other. It is an important programming skill to know how to find the user community and interact with other users to ask for help when you get stuck. As user communities are usually made of volunteers, as your skills improve, you may be able to repay this favor and help new programmers.

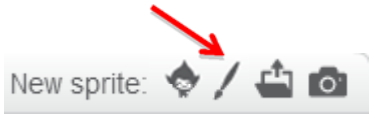
Time to Program with Scratch!

10. Now it's your turn to become a programmer. Open the Scratch homepage and click "Create" at the top of the page to begin a new project. Immediately **name your project** by replacing the word "Untitled" with a name of your choice. **Save your file** by clicking "File" → "Save now".

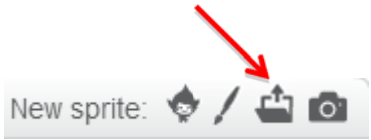
11. As you learned in lecture, in the previous tutorials and the getting started guide, one of the main components to Scratch is the 'sprite'. **Create two sprites of your own**.

COMMON PITFALL: Do not use Scratch Cat, the default sprite. Create your own sprites.

You will use these in your first Scratch project in this section. You can make these any way you like. For example, you could create them by hand in Scratch using the "**Paint new sprite**" button.



However, many more features are available to you if you use GIMP and Inkscape skills from assignment 1. For example, using GIMP you can **extract a new sprite from digital photograph stored on your computer** or found on the Internet. To do so, deleting the background around an object/person in a raster photograph (the same way you extracted your sunflower in Assignment 1), and then save this as a .png file with an alpha/transparency channel. Then use the "Upload sprite from file" button in Scratch to import the file:



If you use Inkscape, note that vector **SVG files** may occasionally produce strange results in Scratch. If that happens, try, saving as "Plain SVG". If that still doesn't work, you can also **convert your vector art to a raster** (bitmap) format by going to the Inkscape File menu and choosing "Export Bitmap". This will create a .png file that you can open into Scratch. (However, you will lose the benefits of vector graphics we discussed in Assignment 1.)

PRO TIP: Scratch will make new sprites from many formats produced by GIMP and Inkscape. It reads the JPG, BMP, PNG, SVG, or GIF image file formats (including animated GIFs).

Also, if you use any image files you find on the internet, be sure to **save a description of the file, the filename,** and the source's **URL** to your lab report HTML document to properly credit your work.

12. This first Scratch project is to get you familiar with "writing"' commands in Scratch and how to use the language. One of the best ways to learn how commands (the colored blocks) work in Scratch is to **right click on a command and select "help"**. This will display a pop-up window with an example of this command being used. Take a look at the "help" for a few commands so you are familiar with how to use this resource.

13. **Drag both of your sprites** up and into the Scratch graphical output window (called the stage) if they aren't already there. This area is where the results of your Scratch program appear.

14. You will now create a very simple computer program. It will consist of series of eight steps, where each step is a Scratch project. Each step/project will build upon the

previous, resulting in a final cumulative Scratch project containing all previous eight parts. With each successive version of this Scratch project you will **save a different Scratch file with a new name and share this new file to the Scratch website for each of the remaining steps in Part A**. You can do this by renaming and clicking "save as a copy". The result will be eight Scratch projects that appear on your user account on the Scratch website, where each version gets successively more functional (that is, it does more stuff).

DON'T PANIC: Although you have to submit eight project files for Part A, the requirements for each one are simple. These are baby steps.

You must name the files differently when you save at each point. For example name them test-1, test-2, test-3, … , test-8 to help you **remember which parts you have completed** and uploaded. These eight iterations of this project correspond to each of the eight categories of "building blocks" in Scratch: Control/Comments, Motion, Sensing, Looks, Operators, Sound, Pen, and Variables. The final project you create in this section must contain blocks from each category.

MORE FUN? For your complete Part A program you can just mess around randomly if you want to. If you prefer a more structured result, you can **set a goal for yourself**. A very simple, interactive one-panel comic is a realistic goal for this part (Part A) of the lab. So is a project that draws abstract art via very limited interactivity, like a keypress or entering a number (think: like a Spirograph). A goal is optional now but later parts will require a more elaborate result.

Visit the following Scratch user: http://scratch.mit.edu/users/stevensondarren. This contains examples of the eight sequential projects that you will be making in the following steps in order to make your first program. **Run each of these** eight programs (test-1 through test-8) **IN ORDER FROM LOWEST TO HIGHEST** by clicking on them and interact with them on the Scratch website. This will help you conceptualize what you will be doing in the following steps of Part A, creating a program and saving a new version with a new file name after each step and then sharing it to your Scratch website gallery. NOTE: These are just simple demos so **there are no Project Notes** to instruct you how to use each program. Instead try things like moving your mouse cursor, clicking and dragging on the screen, clicking on each sprite, etc. to get an idea of how the different programs function.

PRO TIP: You can see the source code of scratch programs. This will be covered later in the assignment, but if you are curious about a program's operation you can also try looking at the source code now – just look for the blue tab that reads "See Inside" on the project page.

15. Now let's get back to working on your own project. Go back to it by clicking on your username, then "My Stuff." Let's begin with the **Events category**. Drag the 'when <green flag> clicked' control statement into the main Scripts window. (Your Scratch projects will frequently contain this "when <green flag> clicked" event statement. It is the

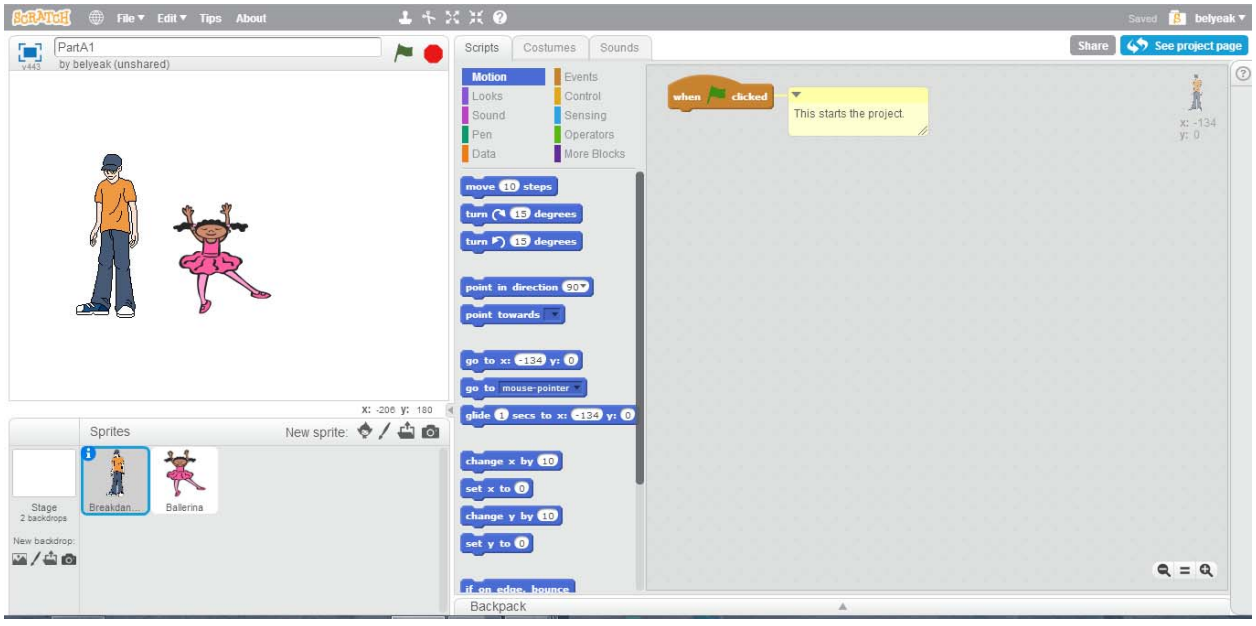
way to make your project execute/begin when a person clicks the green flag icon located in the top right corner of the Scratch project window.)

16. **Add one comment** in the script window of one of your sprites. Remember, if you are unsure how to add a comment or how to do anything else in Scratch, first try to look it up on the Scratch website by doing a keyword search, you will likely find an exact explanation for how to do what you are trying to do.

17. Now you should have a start statement ("when <green flag> clicked"), a comment, and two sprites that are unique to your project. **Save your project** as your first step toward part A (event and comment) and **share it to the Scratch website**. When you are programming in the Scratch application, you can share your project on the Scratch website at any time by selecting the 'Share' drop down menu at the top of the Scratch software interface on your local machine and selecting 'Share This Project Online…'.

COMMON PITFALL: Use a name that includes a "1" so you don't mix it up with the projects you save from the following steps in Part A!

NOTE: Your project is static and does nothing at this point. It should only contain the two sprites you created. The project appears to be just a still image containing the two sprites because you have not commanded (programmed) the sprites to do anything yet.



An example screen shot of a first project file: Two sprites, event statement (orange), and a comment (yellow).

18. Add one or more block(s) from the **Motion** category. Change your **comment** to describe this new code. (TIP: Drag your comment block on top of the code block it most describes and they will appear linked.)

*For the remaining steps in Part A, screenshots from Scratch are included to assist you as you create your first Scratch project. However, if your **final** submission for Part A uses the exact programming blocks (or nearly identical ones) as those used in the screenshot example **you will receive zero points** on this portion of the assignment. It is likely you will use some of the same programming blocks used in the example below, however, there are endless combinations for you to choose from so yours will differ significantly when you do these steps on your own. Again, do not copy this example directly or you will receive no points. These screen shots are for illustration only.*

EXTRA HELP: To get your motion block to do something that you want it to do, you may need to go back to the Control category and add some other control blocks with your Motion block(s) (e.g. wrapping a 'forever' block around your Motion blocks will make them loop), or a 'when <space> key pressed' block to add user input, etc.).

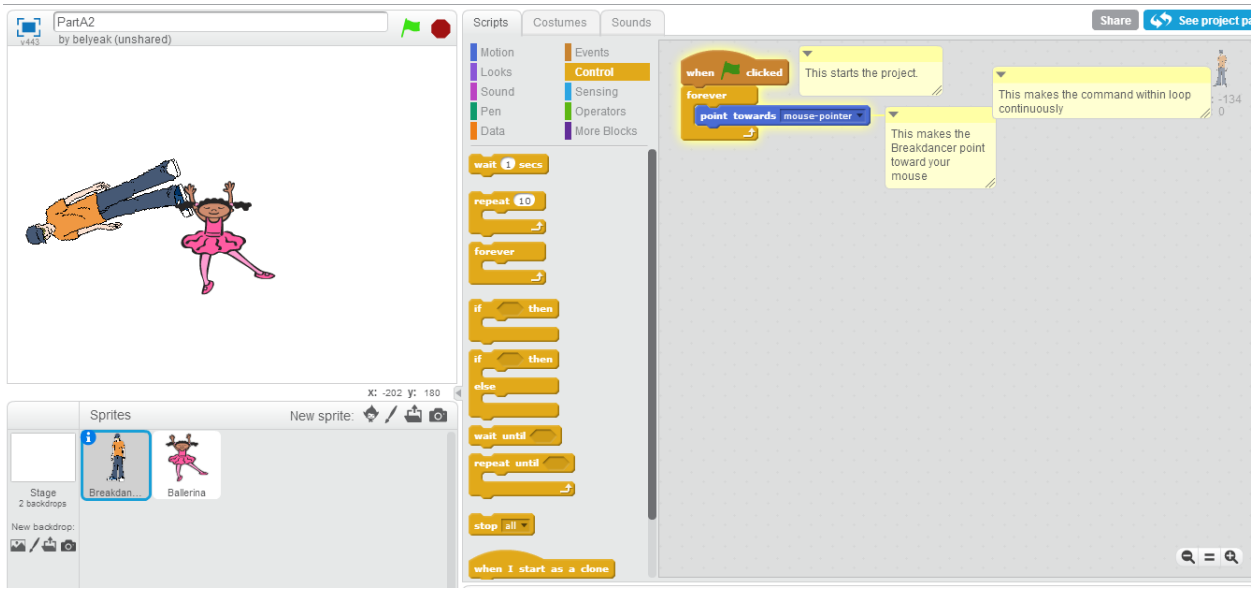
Scripts are automatically **linked to sprites** or to the **entire project**. Notice that you can apply Motion (and all other) blocks to individual sprites by selecting that sprite from the Sprites palette on the bottom right of the Scratch interface (the selected sprite will appear outlined in blue in the list of sprites). You can also add blocks to the overall project by selecting the "Stage" icon first. In this way your Scripts window changes depending on which sprite you have selected, or if you have selected the stage.

COMMON PITFALL: Be sure your script applies motion statements to the sprite you want to move. Check to see which sprite is highlighted by looking at the sprites palette as you work.

PRO TIP: You can drag scripts and drop them on different sprites to copy them to that sprite.

SUPER PRO TIP: Choosing whether your script (code) should apply to just one sprite or to the whole stage is called **scoping**. We'll talk more about that later in the class.

Use "Save as a copy" to **save this** as a Scratch file with a new name including the numeral "2" (e.g. test-2) and **share your new Scratch project** to the Scratch website with "Share". You should now see two Scratch projects in your user account on the Scratch website. If you only see one it is because you are re-saving over the same file each time in which case Scratch will only update your project on the site. You want 8 different projects on the Scratch website in the end, so be sure to "Save as a copy" with a new filename each time before you share.

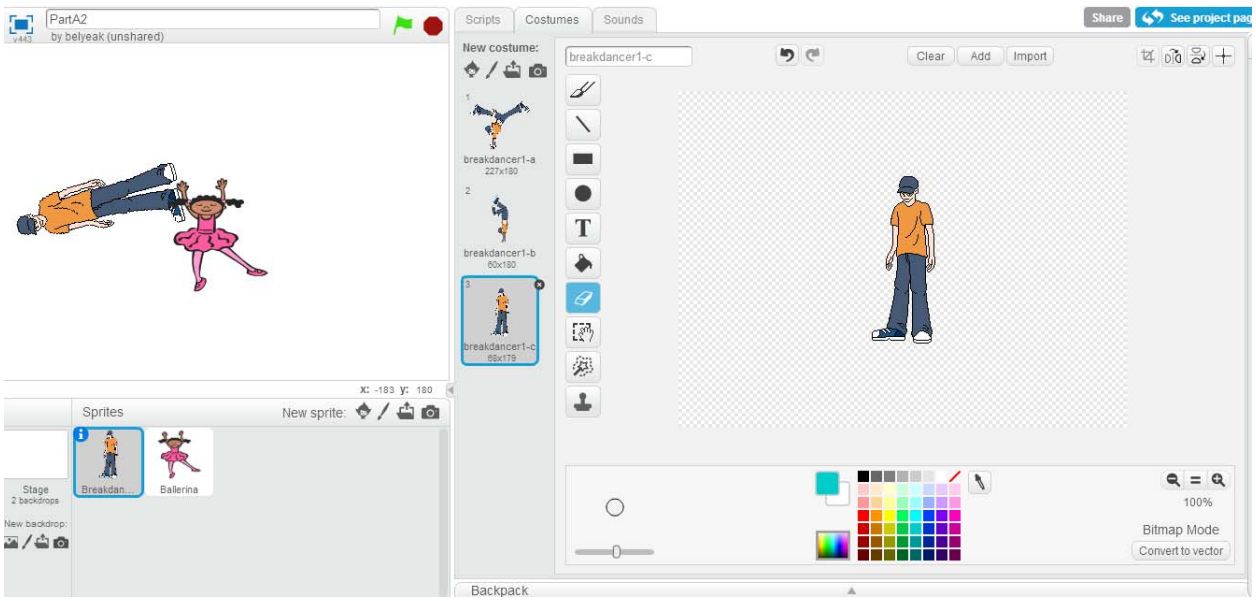


Second project file example: Adding motion statements to a sprite.

19. For your third project file, **add one or more block(s) from the Looks** category. Add one new comment in the script window describing this new code. It is very common to use the Looks blocks to change a sprite's costume. This is one way to produce animation. Explore the "Costumes" tab (shown below) on your own if you want to implement a costume change with Looks statements. Use the Scratch documentation and online resources if you get stuck.

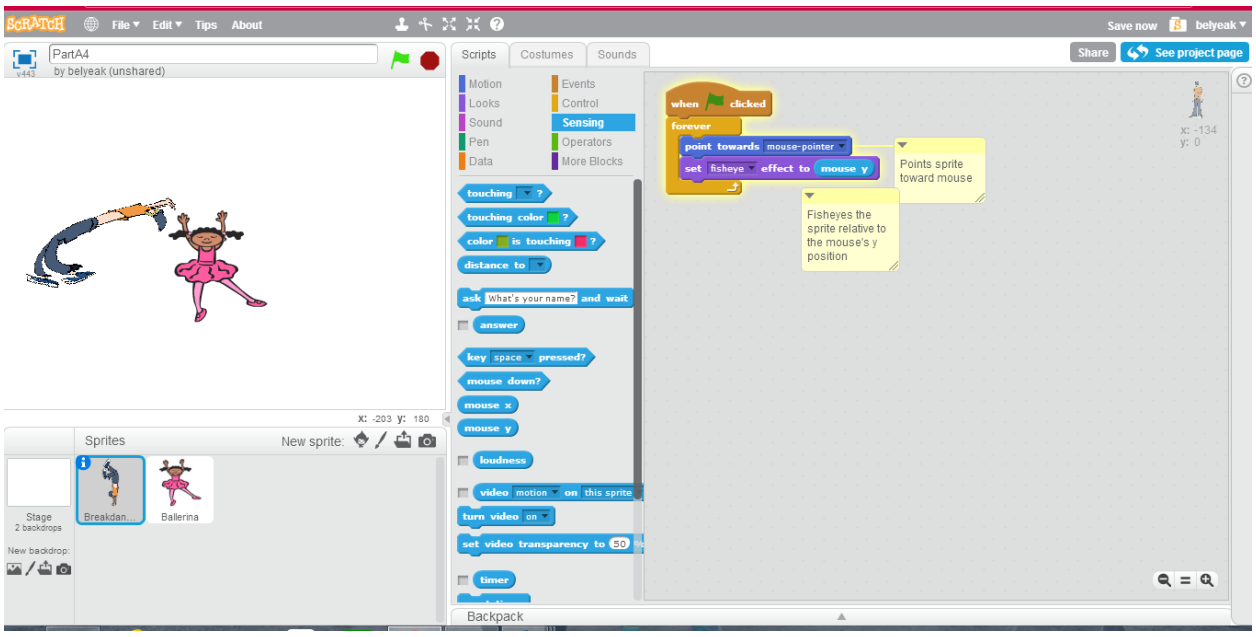
COMMON PITFALL: You must have more than one costume already saved for your sprite in order to switch costumes. You can create costumes for a sprite outside of Scratch or within Scratch using the Costumes tab and selecting the 'copy' option for your original sprite and then editing new costume versions for that sprite.

"Save as a copy" your Scratch project with a new name including the numeral "3" (e.g. test-3) and share to the Scratch website.



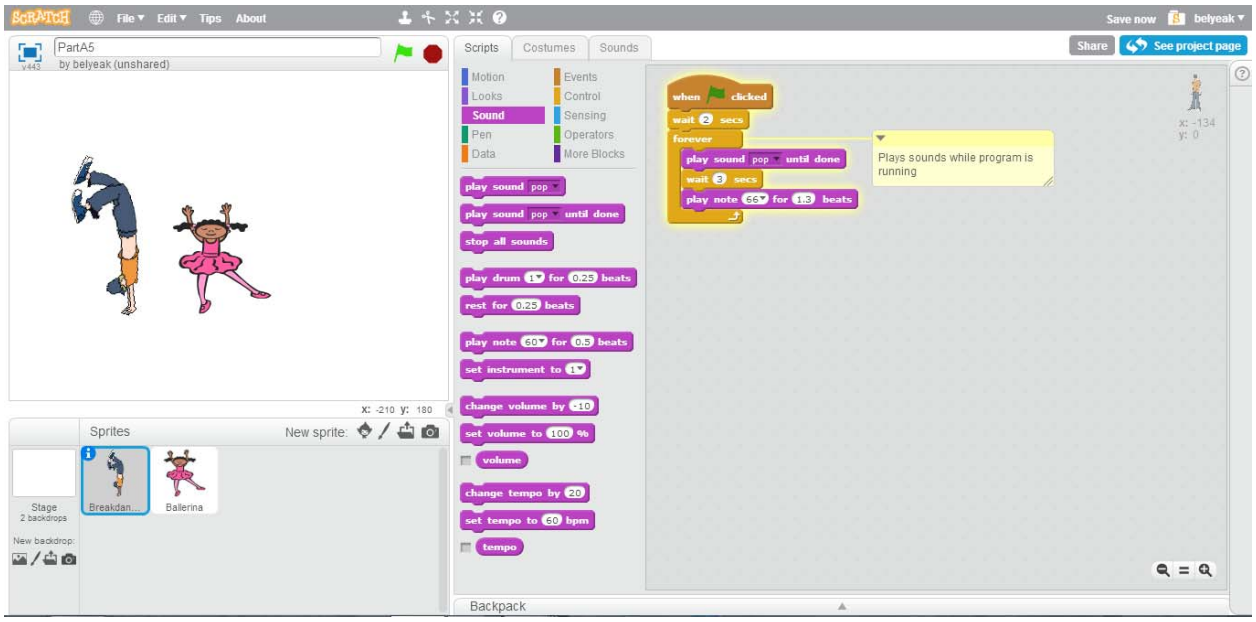
Showing the costumes tab for a sprite.

20. For your fourth project file, add one or more block(s) from the **Sensing** category. Comment your code appropriately. "Save as a copy" your Scratch project with a new name containing the numeral "4" (e.g. test-4) and share to the Scratch website.



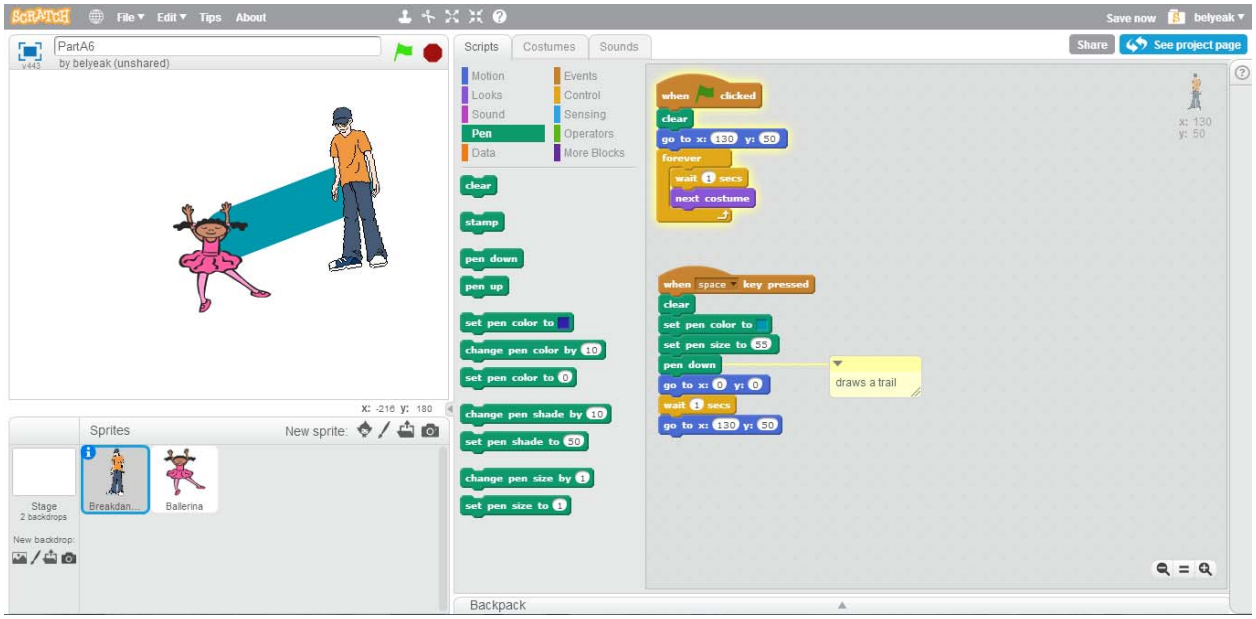
Using a Sensing statement in combination with a 'set (fisheye) effect to' statement.

21. For your fifth project file, add one or more block(s) from the **Sound** category. Comment your code appropriately. Save your Scratch project with a new name including the numeral "5" (e.g. test-5) and share to the Scratch website.



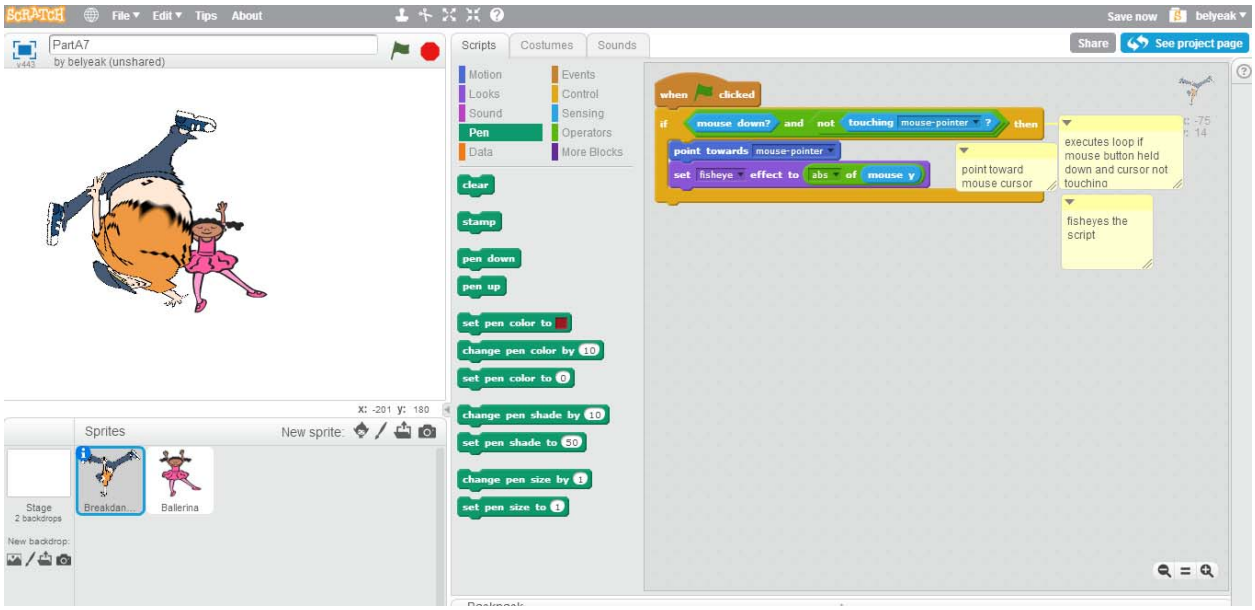
Adding Sound statements.

22. Add one or more block(s) from the **Pen** category. Comment your code appropriately. Save your Scratch project with a new name including the numeral “6” (e.g. test-6) and share to the Scratch website.



Using the Pen blocks

23. Add one or more block(s) from the **Operators** category. Comment your code appropriately. Save your Scratch project with a new name including the numeral “7” (e.g. test-7) and share to the Scratch website.

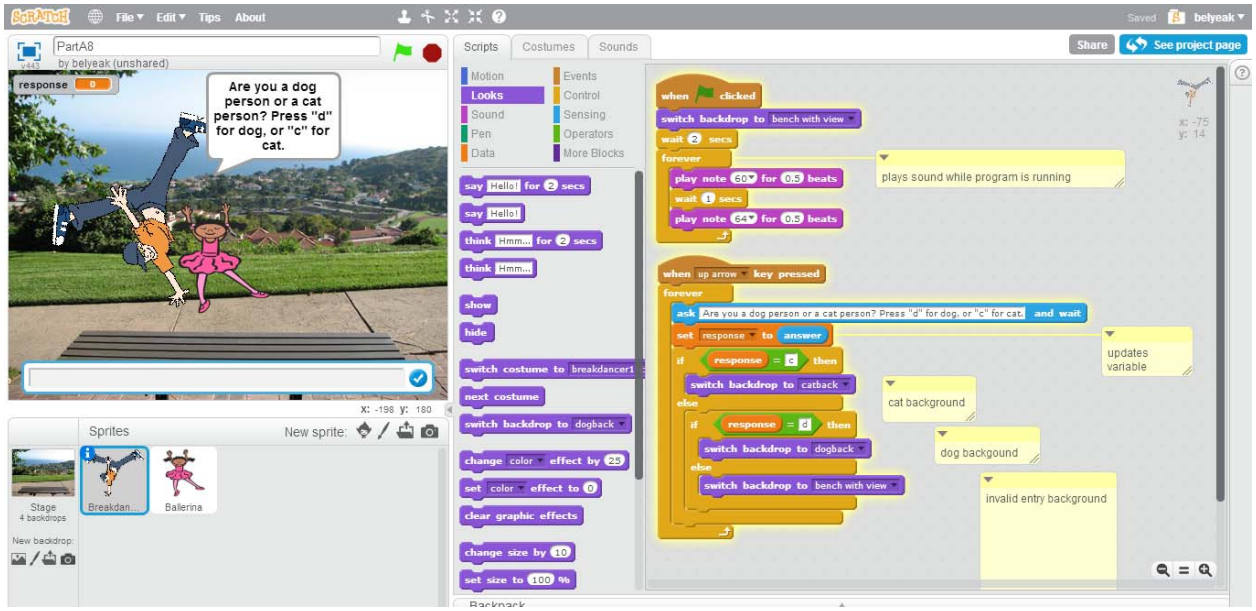


Using Operators.

24. Add a **variable** by pressing "Make a Variable" from the **Data** category, then including one or more blocks from the **Data** category that refer to that variable. Commenting your code appropriately. Save your Scratch project with a new name including the numeral “8” (e.g. test-8) and share to the Scratch website.

    Here is one example of a variable: In test-8 (below) we set the variable ‘response’ equal to our answer from the ‘ask’ block and then, after creating 3 additional backgrounds using the ‘Backgrounds’ tab at the top, we alternate the background file based on the value of the variable ‘response’). Lecture also contained examples of variables. You should use your own variable in a way that you choose.

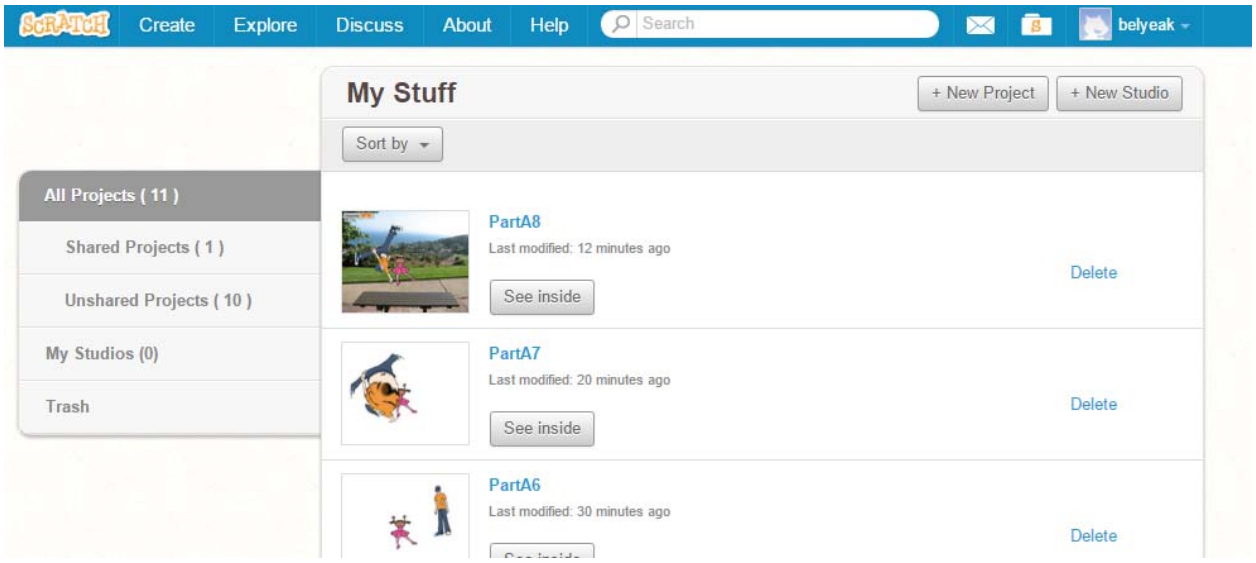
COMMON PITFALL: Commenting your code appropriately means that you also have to name your variables in a comprehensible way so that other programmers (and the graders) can understand them.



Creating and Using a Variable

25. **Add one comment** in the script window that is not linked to a specific block. That is, a comment describing something about the overall program.

26. At this point you should have saved eight different Scratch projects, each with a different name (e.g. test-1, test-2, etc.) and shared each of these to the Scratch website. **Look at your user account** in the "My Stuff" section on the Scratch web site. It should look something like the screenshot below (with different project thumbnails of course):



A Scratch web site user account, under My Stuff.

Each of your eight Scratch projects you just created has a unique URL pointing to that project. One at a time, open each of your eight project pages by selecting the corresponding thumbnail in you're My Stuff and copy the unique URL for each page. **Enter all eight of these URLs into the HTML document** you created in the step A7 (named assignment3.html) and give these links an appropriate heading. Do not use the general URL that goes to your projects gallery. Be sure to use the unique URL corresponding to each program. Each URL will be different. For example, in your HTML assignment3.html file you might format it like this:

```
<h3>Proof that I am programmer (with Scratch)</h3>
<p>
  <a href="http://scratch.mit.edu/projects/stevensondarren/2348463">Test
  1</a>
</p>
<p>
  <a href="http://scratch.mit.edu/projects/stevensondarren/2348467">Test
  2</a>
</p>
```

(…and so on)

Part B: Code Reuse and Remixing

In this section you will create **three more Scratch projects**, however this time you don't have to save and share after each step, and you are not creating them on your own but rather remixing others' code. From this section you will save and share three new Scratch project files and any text responses that are required in your HTML file. Instead of creating a project from the very beginning as you did in Part A, in this section you will modify three existing Scratch projects. The main goals of this section are to get you familiar with **borrowing and modifying code** and to help you begin to think about the interactive Scratch project (game or other interactive project) that you will create on your own in Part C.

FUN FACT: Scratch was named after the sound of scratching in Hip-Hop music. Sampling is fundamental to hip-hop music and the developers of Scratch wanted to encourage code reuse (sampling) as fundamental for the Scratch language. This idea also led to the "Share" command in the Scratch interface. Not all programming languages encourage remixing in this way.

1. If it is not already open, **launch the Scratch website**.

2. Click the link: https://scratch.mit.edu/studios/199905/. These are example games intended to show you basic functions for creating various games using Scratch. One at a time, **open some of the example Games**. Interact with each game and look at the code that was used to create it by clicking "See Inside".

3. After you have explored the eight example games, **open** the Scratch file '**4 Pong'**. It may be found at https://scratch.mit.edu/projects/44790134/

4. Notice in the Pong Scratch file there are two sprites (ball and paddle) and the usual stage. There is also a Sound connected to the ball sprite, located in the Sounds tab at the top when this ball sprite is selected. Using your knowledge from Part A, you will now modify this Pong Scratch file significantly by selecting "See Inside" and then clicking "Remix".

   Before you change the file, use "Save now" to save your (soon-to-be modified) Pong game as a new Scratch file. After you have saved this Scratch file (e.g. pong-remix) share your modified game on the Scratch website as you did in Part A.

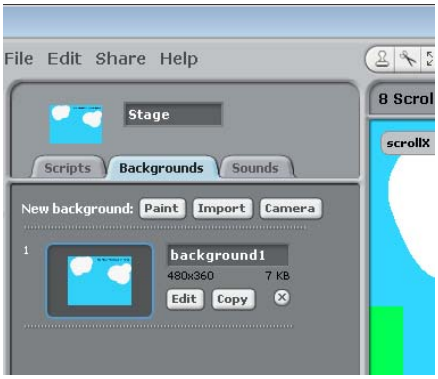
   Easy modifications include re-painting the sprites, uploading entirely different sprites in the place of the ball and paddle, altering background(s), sounds, and especially the behavior of the ball (or other sprite if you swap out ball). Make at least **five significant modifications** (easily noticeable) to the Pong game. Be sure to note the changes as you will be documenting them in the next step. Here is one example of modifying the basic Pong game: http://scratch.mit.edu/projects/Arrowgames75/2263947

5. In your HTML text responses document assignment.html, create a new section and **list the changes you made to the original Pong example game**. Use a list tag like <ul> or <ol> to format your five results. Here you can be general rather than specific. For example, you could include that you "changed the ball sprite" or "modified the ball speed", etc. You do not have to specify exact commands you changed just in general list what you altered, conceptually.

6. Now open the Scratch file **8 ScrollingDemo** at https://scratch.mit.edu/projects/583769/

7. Click "See Inside" and "Remix" as you did in the last part. Save your remix before and after making any changes.

8. This is a 'side scroller' game (similar to the 1980s hit video game *Super Mario Bros*.). As in the previous example, you are going to modify this game. However this time rather than simply picking five or more components to change, you will be changing all of the sprites and the 'stage' background graphic. In addition to the stage there are five sprites, one corresponding to the character (in this case 'player') and four are dynamic backgrounds or 'terrains' (in this case 'terrain0', 'terrain1', 'terrain2', and 'terrain3').



The Backgrounds Tab (with the background sprite selected).

Select the Stage and with this highlighted select the Backgrounds tab (between Scripts and Sounds near top of Scratch software). **Select 'Edit' for this background**. When the Paint Editor window appears edit the Stage background. You can use the built-in tools in the Paint Editor, import your own imagery from GIMP or Inkscape (using the 'Import' button in Paint Editor), or any combination of both. Just make sure you continue to use the background1 file.

EXTRA HELP: Notice how the four terrain sprites also have scripts attached to them that position them correctly. Click on the terrain sprites to see the scripts in the Scripts tab. Observe how they use the variable scrollX and the width of the screen (480) to obtain the correct positioning.

9. Next you will **edit all five sprites** from this example ScrollingDemo game. You must change them completely so that their sizes remain the same but they are not recognizable graphically from their originals. Try to be creative (!) you can make the character and four terrains anything you'd like, however the 'player' sprite is only able to travel on the 'terrains' where he does not touch the color green. You are welcome to change this color in the code. Wherever you apply graphics to the terrain sprites this will change the game completely (create barriers, etc.). Edit each sprite by first selecting it and then selecting its Costumes tab. From there you can select Edit and change the sprite.

PRO TIP: If you have imagery or graphics you make in GIMP or find online, you can import them into this sprite using the 'Import' button and browsing to these files.

10. Once you have changed the stage background and all of the sprites, save the file and **share it to the Scratch website** using "Share". This should be your second shared project from Part B.

11. For this third and last modification example, rather than opening a file from the Scratch examples folder, you will download a Scratch file from the Scratch website gallery and then modify this file using Scratch on your own computer. In a web browser **navigate to the following** page: http://scratch.mit.edu/projects/johndo77/1275591

12. In the upper right part of the screen on this page, **"See Inside" the Sandwich 5.1** file, "Remix" it, and "Save now".

13. Notice the use of the '**Broadcast'** block from the Controls group in this program. If you don't understand the broadcast statement, look up how broadcast works using the Scratch website and other resources, or review the Mahan reading and/or lecture notes.

EXTRA HELP: The **broadcast** block can be very helpful for the game you will design in the final section of the assignment, Part C.

14. Before making changes, use "Save as a copy" to save the Scratch project with a new name with your project files. Then modify the Sandwich 5.1 game by changing any one of the sandwich ingredients. You must change the text icon (e.g. "Ketchup" graphic) as well as its corresponding food 'costume' contained in 'sprite1'. Your new ingredient does not have to be food. You can add a car or whatever you'd like, just **change one ingredient** so that the game **is still fully functional** using your new ingredient. Resave the sandwich Scratch file with a new filename. When you are finished, share this modified Sandwich game to the Scratch website.

DEEP QUESTION: Did johndo77 do a good job documenting their code, choosing appropriate names for **variables** and **messages**? Is the code readable? Is there more that could have been done?

15. In your HTML assignment notes file (assignment3.html), create a section that contains **all three URLs** corresponding to the three modified games you created and shared to the Scratch website in Part B. Give this section an appropriate heading. These three links are the three web page links for each of the three Scratch projects you created in Part B and uploaded to the Scratch website.

COMMON PITFALL: All of the links in your HTML file should work, and they should go directly to the appropriate Scratch project or the TAs won't be able to grade your project. Test them before turning it in!

Part C: Create an Interactive Game

As you noticed in Part B and when browsing the Scratch gallery in Part A, there are many types of games that can be created when programming with Scratch.

Your goals for your game are to make it as user friendly, interactive, bug free, and interesting as you can. The teaching staff must be able to understand your game when accessed on the Scratch website. You may use the "Notes and Credits" or "Instructions" section to include instructions for how to use your game (e.g. "use arrow buttons to move character," etc.).

DON'T PANIC: You don't have to be an artist or a game designer. It is possible to use Scratch's remix functions and borrow graphics from the Web and do well on this assignment.

If you are at a loss for ideas, return to the Scratch Featured Projects to get some inspiration. However, you are (of course) not permitted to copy any Scratch projects from the Scratch website directly and pass them off as your own work. Use other people's code as appropriate for your ideas but **be sure Part C represents an original effort** as well – parts of the code can be re-used but the game must be your own work.

COMMON PITFALL: The teaching staff can see all the Scratch projects that your code borrows from thanks to the "Remix Tree".

With that in mind, you may download and repurpose parts of others' code to help make your own game more functional or elaborate, but the emphasis is for you to create your own unique game using Scratch. If it appears that all of your code is copied directly from another project your Assignment grade will suffer.

Any existing Scratch projects that you include code from must be **explicitly listed in the HTML lab report** file in its own section. Include the URL to the Scratch project you borrowed from as well as a brief description of what component(s) you modified for your own game. For example, you if you were to borrow from the Pong example in Part B, you would list the URL to this Scratch project followed by a specific description of the part you modified and included (e.g. Modified and used block that determines how paddle moves following the mouse).

EXTRA HELP: An old programmer saying is: "Good programmers write great code. Great programmers steal great code." It's meant as a joke, but the message is: sharing code is good. But you must give credit where your work is indebted to the work of others. This is a lot like using quotes and citations in a research paper. Using lots of outside sources is good – that's called research. Using them without attribution is bad – that's called plagiarism and could get you in trouble.

While **you have wide freedom in the type of game you create and its purpose**, there are a few minimum requirements.

EXTRA CREDIT may be awarded if the effort, gameplay, or programming you put into the game impresses the teaching staff.

Create a game (defined as any highly interactive Scratch program) that includes each of the following:

- ☐ **Interactive**. Games are interactive. Users must be able to interact with your Scratch project. Do not make an animation, where the user does not influence the program. Incorporate active input from the user that alters the operation of the program.

- ☐ **Documented and Readable**. Your code must be documented appropriately. (For example, with comments and meaningful file, variable and message names.) At a minimum you must include at least one comment for each sprite you use and at least one comment in your Stage section. As stated above in the intro to this section, you must also list your sources in your HTML project notes file. If you used images from the Web, list your image sources. When published on the Scratch website, include a detailed Notes and Credits or Instructions section informing users how to play your game. Write these notes in a way that any person who can read English and operate a computer could come across your game and easily understand how to interact with it based solely on reading these notes. Aim for extreme clarity. Do not be ambiguous in any way about how to play or use the game, instead use very specific instructions (e.g. “Press the arrow keys (left, right, up, down) to move the Donkey character”, “Click the skull and crossbones button to begin”, etc.)

PRO TIP: Code must be documented and readable to make it maintainable (possible for other people to fix when it breaks, and improve it when necessary.

- ☐ **Bug free**. Users must be able to operate all parts of your game without it crashing or exhibiting unworkable behavior or other hang-ups. Test every portion of your game to be certain there are no bugs and that it is fully functional. If teaching staff cannot run your program and easily plfay your game it is impossible to evaluate (grade). You want it to be as easy to use and as easy to evaluate as possible (e.g. Is it easy to play or use? Does everything work?) Be alert to the problems that might be caused by unexpected user behavior. When you ask the user a Yes or No question (y/n?) what if they press 9?

- ☐ **Scripts Must Serve Your Project's Goals**. In this Part C game, it is important that the statements you write in Scratch make sense in the context of your project. For example, your Part A projects did not make sense – they were just a series of scripts that (while they worked) did not do anything useful. Part B's scripts *did* make sense, but they involved only slight changes to someone else's game concept. For Part C, you can not receive an "A" by simply inserting Scratch statements in a way that (while it runs) does not make any sense. Your code should serve your project, and we will judge this by playing your game and reading your comments.

- ☐ **Your Own Sprites**. Include at least five original sprites that you create and/or import into Scratch. GIMP and Inkscape are excellent for making sprites (but be sure to export to bitmap if you use Inkscape), or use built-in Scratch tool called 'Paint New Sprite'. Additionally, you can find images online and import these as sprites. In this case "original" only means not the sprites that are already built into Scratch in the sprite library and not those taken from others' Scratch projects. It is OK to modify graphics from the Web, but as stated above in the intro to this section, you must also list your sources in your HTML project notes file. If you used images from the Web, list your image sources. Be creative in your sprites!

- ☐ **Audio**. Your game must utilize audio. Any number of sounds is fine, the minimum is one.

- ☐ **List Data**. You must include at least one **list of variables** in your project using the "List" commands. If you aren't sure what lists are, review your lecture notes, the Mahan reading, and/or the documentation resources described above. Note that it is OK to "hide" the list (by right-clicking on it). It does not need to appear on your stage when your program is running.

EXTRA HELP: If you hide a list from the stage you can make it appear again by checking the box next to the list's name in the "variables" pane where the scratch blocks are kept.

- ☐ **Teach Yourself One Additional Scratch Concept**. Use one of these statements in your code, subject to the other rules above (well-documented, makes sense, bug-free, etc.):
  - **Cloning.** Clone a script in a useful way. http://wiki.scratch.mit.edu/wiki/Cloning
  - **Broadcast**. Send messages between scripts using "broadcast" and "when I receive". http://wiki.scratch.mit.edu/wiki/Broadcast
  - **Timer.** Use the built-in "timer" variable under "sensing" to control something in your game. (Note that this is not the same as the date/time variables.) http://wiki.scratch.mit.edu/wiki/Timer
  - **Webcam.** Use the "turn video on" block under "sensing" and then do something with the video information (employing the video variables "motion" and/or "direction") to incorporate a Webcam's input into your game.
  - **Microphone.** Use the "loudness" built-in variable under "sensing" to use your computer's microphone to add interactivity to your game.
  - **Custom Block.** Make your own custom block. Making a custom block that is useful and makes sense may be challenging if you are not familiar with programming. http://wiki.scratch.mit.edu/wiki/Custom_Blocks

After you have completed your game using Scratch for Part C, share it on the Scratch website and **include the URL to this project in your HTML assignment notes** file in a new section:

```
<h3>Game for Part C</h3>
<p><a href="http://scratch.mit.edu/projects/stevensondarren/431">Rabbid
Rabbit Revenge</a></p>
```

Be sure to test your game thoroughly on the Scratch website to be sure it is completely functional and bug free before you turn it in.

THE FINAL QUESTION: CRITIQUE A PROJECT

Finally, answer the following question at the bottom of your HTML lab report file. In one paragraph of at least 200 words, discuss how someone else's Scratch project does not meet one or more of the "good programming" principles discussed during lecture and describe what you would do to improve it. You can find a new script to critique, you could critique one of the Scratch projects you chose as an example, or you could critique one of the Part B scripts you were asked to modify. Include a link to the project you are critiquing and make sure your criticisms are clear. What should be done to improve the scripts? Please spell-check and proofread your answer. If it helps you can remix the script and actually edit it, but please still explain your problems with it in clear English at the bottom of this lab report HTML file. Documentation and commenting are important, but BE SURE YOU ADDRESS SOME SUBSTANTIVE CONCERN BEYOND just commenting and documentation.

Refer to the "**Turn it in**" at the beginning of this document to turn in your completed work.

</assignment>

BONUS WORK:

Lab assignment 3 is over, but if you are interested in learning more about Scratch (or programming), or the assignment was too easy for you, you can receive extra credit for bonus work. For this assignment, there are a few options for bonus work. You can decide how to demonstrate you have completed the bonus work. For instance, you might share a URL, upload

an additional HTML file, add a screenshot, and/or write a very short note about what you did at the end of your lab report.

For this assignment each bonus work task can receive up to the amount indicated of extra credit on this assignment, up to a maximum of +40%. Here are the options:

- **Amazing Game, Amazing Effort, Amazing Code**. If the effort you put into the game impresses the teaching staff, or if the gameplay or programming impresses the teaching staff, you may receive extra credit. (+% varies)
- **Better Pong.** The physics of the Pong game you were asked to modify in Part B really suck. But it is possible to make a much more realistic bouncing ball in Scratch. Improve the physics of the bouncing ball. (+% varies)
- **Embed Scratch**. You can embed a Scratch project using the "Embed" button at the bottom of the editor. Embed this project in your Web page rather than using a link for (+0.5%).
- **Highlight Reel**. Use "File" > "Record Project Video" to make a highlight video showing off your game. Upload it to YouTube and embed it in your HTML file for this assignment. (+1%)
- **Learn More Concepts**. Correctly use more of the additional concepts in Part C listed after "Teach Yourself One Additional Scratch Concept." (+3% for each concept)
- **Executable Standalone Scratch**. It is possible to distribute your Scratch game as a standalone program, although it can be a pain. This is explained here: http://wiki.scratch.mit.edu/wiki/Porting_Scratch_Projects but much of this information is out-of-date and incomplete, so this will involve some additional research on your part. To receive credit you must convert your Part C Scratch 2.0 project into a standalone .exe (Windows) or .dmg (Mac) file so that it will execute without a Web browser or Flash player. This will be difficult if you are not comfortable with computers, converting files between formats, and independent research. (+10%)
- **Advanced Programming (1).** Use recursion. This will be difficult if you are not familiar with programming. http://wiki.scratch.mit.edu/wiki/Recursion#Recursion_in_Scratch_2.0 (+5%)
- **Advanced Programming (2).** Simulate a multidimensional array. This will be difficult if you are not familiar with programming. http://wiki.scratch.mit.edu/wiki/Array#Multidimensional_Arrays_in_Scratch (+5%)

IMPORTANT: If you are submitting bonus work, be sure you make a note of this on the bottom of your **lab report** HTML file so that you receive credit for it.

tl;dr