FIRIA LABS

LESSON: CodeX Egg Hunt Project (April)

Time: 2 class periods

Overview:

This project combines pixel art with a custom module while learning about digital easter eggs. This project is designed as a class project, and not for individuals or a self-paced lesson.

Introduction: Learn about digital easter eggs by looking at some in Wikipedia and Google.

Part 1: Students design their own decorated egg as pixel art. You can print a paper grid for students to use while designing their egg. The grid is 24x24, which is the same size as all the built-in images and is easy to scale. You can have students build their pixel art from scratch in CodeSpace, or you can provide a template. All images are combined into a single file for sharing.

Part 2: Students upload their custom module of pixel art. Then they create a program that "hides" surprise images behind their decorated eggs as a simulation of digital easter eggs.

Several ideas for extensions are given in the last section of slides. These are optional, and they are geared toward students with some programming experience. Some sample code is given for some of the extensions, but no specific instructions. Sample code as text files are provided in teacher resources.

Coding Objectives:

- I can design a pixel art image.
- I can program an original pixel art image.
- I can create a custom module of pixel art images.
- I can import a custom module and use the images it contains.
- I can program a button to display a random pixel art image from the custom module.
- I can append an item to a list.
- I can check if an item is in a list.
- I can remove an item from a list.
- I can determine if a list is empty.

Grades 6-8 CS Standards:

2-CS-03 Systematically identify and fix problems with computing devices and their components.

2-AP-10 Use flowcharts and/or pseudocode to address complex problems as algorithms.

2-AP-11 Create clearly named variables that represent different data types and perform operations on their values.

2-AP-12 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.

2-AP-13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-16 Incorporate existing code, media and libraries into original programs, and give attribution.

Grades 9-10 CS Standards:

3A-CS-03 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.

3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.

3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.

Grades 11-12 CS Standards:

3B-AP-10 Use and adapt classic algorithms to solve computational problems.

3B-AP-14 Construct solutions to problems using student-created components, such as procedures, modules and/or objects.

3B-AP-16 Demonstrate code reuse by creating programming solutions using libraries and APIs.

3B-AP-22 Modify an existing program to add additional functionality and discuss intended and unintended implications.

Preparation:

- Download slides.
- Check the linked websites for easter eggs.
- Download and print the egg template for each student. Print the blank template as needed for surprise images.
- Be familiar with the final code.
- Read through the teaching guide.
- Optional: Use the accessibility guide as needed.

In the folder:

- Egg Hunt project slides
- Egg template
- Standard template
- eggs.py template code
- EggHunt code solutions for final, extension1, extension2, extension3 and all extensions.

Agenda:

- Warm-up (15 minutes)
- Part 1: Pixel art (30 minutes)
- Part 2: EggHunt code (45 minutes) – could be longer with extensions

Teacher Notes:

Part 1

- Each student should design and program a decorated easter egg using the template. This will keep all the pixel art images the same size and easy to work with in the code. You can give each student the 24x24 paper grid.
- You will need a few extra images for the hidden surprise (digital) easter eggs. You can handle this several ways:
  - Have a few students design the surprise images instead of an easter egg.
  - Ask a few students who work quickly to design a decorated egg and a surprise image. They can be in the same program code.
  - Use pixel art that was created earlier. With this option, it is best if the pixel art is also 24x24.
  - You can create a few surprise images for students to discover.
- The students should submit their pixel art programs to you so you can compile all the images into a single file. The file doesn't need to have any additional code, other than the two lines of code shown in the slide. Any code that displays an image should be deleted. Otherwise the images will be displayed when the custom module is imported.
- Save the compiled images as a text file and give it back to the students. When you compile the images, make sure no code for displaying images is used, and make sure you include the two lines of code shown slide 18.

Part 2

- The program uses several lists. It is helpful if your students have already completed Mission 6-9. Review creating and using lists as needed.

Other Extension Ideas:

- Use an easter egg image in the Holiday Tag project (December).
- Use JPG images instead of pixel art for the hidden easter eggs.
- Use text messages for the hidden easter eggs.
- Keep track of the lowest number of clicks needed to find all the easter eggs and display in the ending.
- For the quilt pattern, try a different scale.
- Add an easter egg to one of your programs.
- Use your creativity and come up with other possibilities for the program.

Cross-curricular Connections:

- **LANGUAGE ARTS:** Do more research on easter eggs. Write a report.
- **MATH:** This project uses probability. Have a lesson on probability and do other fun unplugged activities with probability. Calculate what the probability should be of any given image, and then run experiments to find the actual probability. Make a chart.
- **VISUAL ARTS:** Design more pixel images and create another custom module of pixel art to upload and import.

Teaching Guide

Warm-up (15 minutes)

Slides 2-9

This project is a fun way to use easter eggs as a backdrop for learning about digital easter eggs and creating a simulation.

- Go through the slides about digital easter eggs.
- The first example is in Wikipedia.
- The next examples are in Google.
- If your students do not have access to the websites, consider showing these to the class on a large screen.

💻 This project should be completed as a class, and not individually or self-paced. Students should already have an account in CodeSpace.

💡 **Teaching tip:**

The instructions for the project are not included in the interactive textbook of CodeSpace. Download and follow the slides. They include step-by-step instructions as well as code segments to guide students through the program code creation.

Day 1: Design and Code a Pixel Image (30 minutes)

💻 **Slide 11**

Give students a 24x24 paper grid for the egg template. They can use color pencils or markers to design their decorated egg. Students will write down the name of their egg. To make coding easiest, assign each student a number for their egg, like egg1, egg2, etc. Students can also use their name as the name of their egg. During the list creation, students need to know the names of the eggs and the surprise images.

Decide how you will get the surprise images that will be "hidden" behind the decorated eggs. If you are having students design the surprise pixel images, give them a 24x24 paper grid without an egg outline to use for their design.

💻 **Slides 12-15: Step 1 & 2**

Students create a new file in CodeSpace and program their pixel art. The slides don't go into a lot of detail here; hopefully your students have created pixel art before this assignment. If not, this part may take longer.

You can either:

- Have students start from the beginning and type all the code (follow the slides) –OR–
- Give students the egg template code. It is available with just the 24x24 grid, or with the grid and egg outline.
- Use the method that works best for your students.

Things to remember:

- You can create one line of 24 periods and then copy and paste it 23 more times.
- The spacing is very particular when creating pixel art.
- You cannot include spaces around the equal sign when creating the color legend.
- Use the standard built-in colors, or use an RGB color picker and use a custom color.
- If you are using a custom color, there cannot be spaces around the commas.
- Make sure each line of code is 24 characters long.

💻 **Slide 16: Step 2**

Students add two lines of code to make sure their image displays correctly on the screen. If students create more than one image, they should test each image. This code will be deleted after testing.

💻 **Slide 17: Step 2**

Students submit their pixel image to the teacher. Decide how you will collect the images. You can:

- Have each student download their code and send it to you.
- Create a google doc and have each student copy and paste their code into the document.
- Have students submit their code through your LMS.
- Any way that works for you.

💡 **Teaching tip:**

After you have all the images, compile them into a single program. This can be done in CodeSpace so you have debugging capabilities. The file should include all the surprise images as well as all the decorated eggs. All pixel art! **Do not** include any code that draws an image on the screen:

```
draw_egg = ascii_art.get_images(found_egg)
display.draw_image(draw_egg[0], scale=10
```

Do include this code at the end of the program:

```
if __name__ == '__main__':
    print("Loaded eggs module")
```

Download the file as a text file.

Day 2: Code the Egg Hunt program (45 minutes)

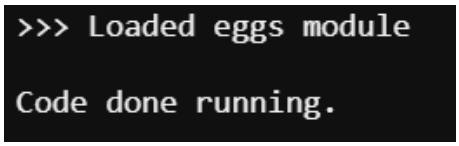
💻 **Slide 18: Step 3**

Share the compiled text file with your students. They will copy the code and paste it into their egg.py file.
Note: It is very important that the file have the extension .py!!

It is okay if they overwrite their original file with the compiled code. If they want to keep their original file, they need to do a *save as* before pasting the code. Their compiled code still needs to be the **eggs.py** file.

💻 **Slide 19: Step 3**

Run the code. Nothing shows on the CodeX. If students open the console, they should see the message:



💻 **Slides 21-24: Creating lists**

Students start a new program, import modules, and create three lists. The items in the first two lists come from the compiled eggs program. Students need to know the names of the eggs and the surprise images. You can have them scroll through the egg file to see the names, or you can make a list of the egg names and the surprise names.

💻 **Slides 25-26: Display a random egg**

The steps to the program are broken down into specific steps, each with an algorithm presented in pseudo code. If your students have experience, have them try writing the code from the algorithm. The second slide has the code.

The first step is to display a random egg from the eggs list, using a button press to initiate the event.

💻 **Slides 27-28: Hide the surprise images**

Students use a for loop to add items to the empty hides_an_egg list. This code segment may be new for students, using the append function for loops.

💻 **Slide 29: Find the easter eggs**

This slide introduces two list functions they will use in the next code, and gives an example.

💻 **Slide 30: Find the easter eggs**

This slide gives the algorithm for finding the easter eggs. Experienced students can try doing the code by themselves, but the code is given on slide 31 if needed.

💻 **Slides 31-32: Find the easter eggs**

Sample code is given, and instructions for testing and debugging.

💻 **Slides 33-35: All eggs found!**

The last step. Write code that displays a message once all surprise images have been found. The program can continue to display the decorated eggs, but no need to check for easter eggs.

You can stop here, or continue with extensions. All extensions are optional.

💻 **Slides 36-43: Extensions**

Several ideas are given for students who want to challenge themselves and apply their learning.

Specific instructions are not given, but you can work on them together as a class or students can work individually. Code solutions are given.

Extensions #4 use the pixel images in a different way. The code requires nested loops. It is a good activity for experienced programmers. Sample code is given in the slides for this extension.

Wrap-up / Optional

💻 The project doesn't require a wrap-up. You can use an extension or extra-curricular activity from the list above, or have students reflect on the project and something they learned.