Lesson 14: Mini-Project - Animation

45 minutes

Overview

This lesson is a chance for students to get more creative with what they have learned as students are asked to combine different methods that they have learned to create an animated scene. Students first review the types of movement and animation that they have learned and brainstorm what types of scenes might need that movement. They then begin to plan out their own animated scenes, which they create in Game Lab.

Question of the Day: How can we combine different programming patterns to make a complete animation?

Standards

Full Course Alignment

CSTA K-12 Computer Science Standards (2017)

- ▶ **AP** - Algorithms & Programming

Agenda

Warm Up (5 minutes)
- **Review**

Activity (35 minutes)
- **Step 1: Define - Describe Your Scene**
- **Step 2: Prepare - Design Your Scene**
- **Step 3: Try - Develop Your Scene**
- **Gallery Walk**

Wrap up (5 minutes)
- **Journal**

Preparation

- Check the **"Teacher's Lounge"** forum for verified teachers to find additional strategies or resources shared by fellow teachers
- If you are teaching virtually, consider checking our **Virtual Lesson Modifications**

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the teachers

- **Mini-Project - Animation** - Slides
  - Make a Copy

For the students

- **Animated Scene** - Rubric
  - Make a Copy
- **Animated Scene** - Activity Guide
  - Make a Copy
- **Problem Solving with Programming** - Resource
  - Make a Copy

Teaching Guide

Warm Up (5 minutes)

Review

Prompt: Write down as many types of movement and animations as you can remember from the previous lesson. Make sure you know what blocks and patterns you need to make those movements, and when those movements would be useful.

Share: Allow students to share out what they remember as a group review.

Discussion Goal: The goal of this discussion is to review the different types of movement and animations that students have learned. As students talk about the different methods, make sure that they are mentioning why that type of movement would be useful. Press them to be specific about what they might animate using the various methods.

Remarks

> Now that we can control the way that our sprites move a little better, we're going to have a chance to put everything together to make an animation from scratch.

Question of the Day: How can we combine different programming patterns to make a complete animation?

Teaching Tip

Facilitating Mini-Projects: Mini-Projects act as checkpoints in the curricula and cover the subset of skills students have seen so far in the unit. They are designed for 1-2 days of implementation as a way to check-in with how well students understand the course content so far. You may decide to extend these projects as a way to support or challenge students, which could allow you to revisit difficult concepts or support students who may have missed lessons and are trying to catch up. However, we recommend deciding this ahead of time and being firm with students about how much time they have for each project - otherwise, it's easy for projects to drag-out to multiple days and for student's work to spiral beyond the scope of this project.

Activity (35 minutes)

Distribute: Pass out copies of the **Animated Scene**. Students should use this worksheet to guide them through the Problem-Solving Process and plan out the animated scene they create at the end of this lesson.

Teaching Tip

Facilitating Group Projects: If students are working in pairs or small teams to complete projects, consider showing these two videos to the class:

- **How Teamwork Works**
- **Dealing with Disagreements**

Depending on your goals with this project, consider having teams complete a **Student Guide to Team Planning**, which reinforces the message in the video

Optional Transition You can choose to either send students to Code Studio to see an example of an animated scene in level 1 or you can show the example in the slides.

1 **Example Animated Scene**

Step 1: Define - Describe Your Scene

Circulate: As students fill out a description of the animated scene they want to create, circulate the room to ask students about their ideas and help guide them to make sure they are thinking of how they could incorporate movement in a scene with the draw loop.

Step 2: Prepare - Design Your Scene

As students answer questions about the design of their scenes, continue to ensure that they are thinking about movement and using the draw loop. After they have completed their designs, ensure that they can identify which sprites will be needed and what sprite properties they need to animate for desired movement in their scene.

Teaching Tip

Scoping Student Projects: Students may ideate projects that are beyond the skills they currently have or that would take longer than the allotted time to implement. Rather than asking students to choose a different project, consider asking students to imagine a more scaled-down version of their initial idea. As an analogy, if students initial idea is the "Run" step, imagine a less intense version that represents what the "Walk" step would look like. If necessary, you can keep going back further to a "Crawl" step as well.

Digging Deeper: This is sometimes referred to as the Minimal Viable Product - you can learn more about this process and adapt it into your project strategies by reading this article: **Making Sense of MVP** by Henrik Kniberg

Step 3: Try - Develop Your Scene

Distribute: Hand out a copy of the **Problem Solving with Programming** to pairs of students or have them take this resource out if they still have it after the last project. Encourage students to look over the guide.

Display: Show the slide with the problem solving process graphic.

Remarks

> Many of you are ready to start creating your programs. Don't forget about using the problem solving process to help with the *blank screen* effect. If you feel stuck or you're not sure what to do next, remember you can always follow the steps of the problem-solving process to **define** your next step, **prepare** for what you want to code, **try** it out, then **reflect** on whether or not it solved your problem.

Circulate: Encourage students to use the steps in the Problem-Solving Process for Programming when they get stuck or are unsure of what to do next.

Teaching Tip

Not all bullets in the Problem Solving Process will be applicable to every problem a student has. Instead, encourage them to pick one or two from each category to try each time they are stuck

After you have checked the designs, allow students to log into Code Studio and code their programs.

Transition: Send students to Code Studio.

2 **Draw a Background**

Teaching Tip

Scaffolded Tasks: The tasks for this mini-project are broken down for students with each level focusing on a particular part of the project (background, sprites, text, etc). Encourage students to follow the instructions in each level, focusing on one task at a time.

3 Add Sprites

Teaching Tip

Debugging Strategies: As students design and implement their own project ideas, they may find themselves with new bugs that they need to untangle and you may find yourself looking at completely unfamiliar code as students look for help troubleshooting their errors. To help smooth out the debugging experience, consider the following strategies:

- Review the **Teacher Guide to Debugging** for some common questions and strategies to help support students in debugging their code
- Have students follow the steps in the **Student Guide to Debugging** and use the **Bug Report Quarter-Sheets** as an initial step in the debugging process. This helps students prepare and communicate their issue before asking for help.
- If students haven't seen it yet, consider showing the **Debugging Video** to the class to reinforce debugging best practices.

Digging Deeper: Consider supplying students with an object to talk to as part of the debugging process. This is sometimes known as Rubber Duck Debugging - you can learn more on the website **https://rubberduckdebugging.com/**

4 Add Text

5 Add Movement

6 Review Your Animated Scene

Gallery Walk

Allow students to walk around the room and see the pictures that each of their classmates has coded. Celebrate all of the different ideas that students were able to implement with the same basic code.

Teaching Tip

You may choose to formalize this process by having each student write down one positive quality of each project, or having students "draw names" to comment on one particular classmate's work.

Assessment Opportunity

Use the project rubric attached to this lesson to assess student mastery of learning goals.

Wrap up (5 minutes)

Journal

Question of the Day: How can we combine different programming patterns to make a complete animation?

Prompt: What was one interesting way that you saw sprite movement used today?

Share: Have students share out what they appreciated about their classmates' projects. You may want to do this "popcorn" style, with each student who responds choosing the next person to share.

Discussion Goal: This discussion should serve as a celebration of what the students have accomplished. As students share out what they have seen, encourage them to learn from each other and ask questions if they were not sure how to do something. Highlight how students were able to do very different things with the same tool.