

Lesson 18: Velocity

Overview

Question of the Day: How can programming languages hide complicated patterns so that it is easier to program?

After a brief review of how they used the counter pattern to move sprites in previous lessons, students are introduced to the idea of hiding those patterns in a single block. Students then head to Code Studio to try out new blocks that set a sprite's velocity directly, and look at various ways that they are able to code more complex behaviors in their sprites.

Purpose

This lesson launches a major theme of the chapter: that complex behavior can be represented in simpler ways to make it easier to write and reason about code.

In this lesson students are taught to use the velocity blocks to simplify the code for moving a sprite across the screen. This marks a shift in how new blocks are introduced. Whereas previously blocks were presented as enabling completely new behaviors, they are now presented as simplifying code students could have written with the blocks previously available. Over the next several lessons, students will see how this method of managing complexity allows them to produce more interesting sprite behaviors.

Assessment Opportunities

1. **Use the velocity and rotationSpeed blocks to create and change sprite movements**

See Level 9 in Code Studio. (Level 5 can be used to check rotationSpeed.)

2. **Describe the advantages of simplifying code by using higher level blocks**

In the wrap up, check students' descriptions of the blocks that they would create and why they would want to create them.

Standards

Full Course Alignment

CSTA K-12 Computer Science Standards (2017)

► **AP** - Algorithms & Programming

Objectives

Students will be able to:

- Describe the advantages of simplifying code by using higher level blocks
- Use the velocity and rotationSpeed blocks to create and change sprite movements

Links

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

For the teachers

- **CSD Unit 3 - Interactive Animations and Games** - Slides
- **Velocity** - Resource

For the students

- **Velocity** - Video (**Download**)

Introduced Code

- rotationSpeed
- velocityX
- velocityY

Agenda

Lesson Modifications

Warm Up (5 minutes)

Activity (35 minutes)

Wrap Up (5 minutes)

Journal

Teaching Guide

Lesson Modifications



Attention, teachers! If you are teaching virtually or in a socially-distanced classroom, please **click here** to access modifications that can be used during this lesson.

Warm Up (5 minutes)

Demonstrate: Ask for a volunteer to come to the front of the class and act as your "sprite". Say that you will be giving directions to the sprite as though you're a Game Lab program.

When your student is ready, face them so that they have some space in front of them and ask them to "Move forward by 1". They should take one step forward. Then repeat the command several times, each time waiting for the student to move forward by 1 step. You should aim for the repetitiveness of these instructions to be clear. After your student has completed this activity, have them come back to where they started. This time repeat the demonstration but asking the student to "Move forward by 2" and have the student take 2 steps each time. Once the student has done this multiple times ask the class to give them a round of applause and invite them back to their seat.

Prompt: I was just giving instructions to my "sprite", but they seemed to get pretty repetitive. How could I have simplified my instructions?

Discuss: Give students a minute to write down thoughts before inviting them to share with a neighbor. Then have the class share their thoughts. You may wish to write their ideas on the board.

 Discussion Goal

Goal: The earlier demonstration should have reinforced the fact that repeatedly giving the same instruction is something you would never do in real life. You would instead come up with a way to capture that the instruction should be repeated, like "keep moving forward by 1."

Remarks

Programming languages also have ways to simplify things for us. Today, we're going to look at some blocks in Game Lab that hide complicated coding patterns to make things easier for programmers.

Question of the Day: How can programming languages hide complicated patterns so that it is easier to program?

Activity (35 minutes)

Remarks

One way to simplify these instructions is to just tell our "sprite" to keep moving by 1 or 2, or however many steps we want. As humans, this would make instructions easier to understand, and as we're about to see there's a similar way to make our code simpler as well.

Transition: Move students to Code Studio

Circulate: These levels introduce the velocityX, velocityY, and rotationSpeed properties that you just discussed with students. Check in with students to see how they are doing and keep track of when everyone has made it to the end of level 10.



Skill Building

Questions to Consider with Video:

- Why might you want to use a velocity block instead of the counter pattern?
- Give an example of a counter pattern and how you could use a velocity block instead.



Video: Velocity

Discussion Goal

Discussion Goals

It may not be obvious to students why the velocity block is so powerful. The immediate answer is that the velocity block allows a programmer to set the velocity at the beginning of the program and not have to worry about the counter pattern inside the draw loop (as Game Lab will take care of that). If students are having trouble thinking of situations in which the velocity block provides a big advantage, assure them that they will tackle some problems in the coming lesson that they will need this block for.

As students give you examples, try to elicit answers that use both positive and negative numbers, and that use the x and y positions as well as sprite rotation.



Skill Building

3

4

5

6

7

Teaching Tip

Inside versus outside the draw loop

This is a good time to remind students that code outside the draw loop is used to set up the program. It is for how you want your program to start. Code inside the draw loop is for things that are changing as the program is running, user interaction.

There may be some confusion that the new blocks are animation (changing position) and yet have gone outside the draw loop up until this point. That is because up until this point, the velocity has been set at the beginning of the program and not changed. When students want the velocity to change during the program, it will need to go inside the draw loop.



8

Practice



9

Assessment

✓ Assessment Opportunity ▲

You can use this level as a formative assessment for students. Click inside the level to view a rubric and leave feedback to your students



10

Challenges

Wrap Up (5 minutes)

Journal

Prompt: You learned a few new blocks today. At first glance, these blocks did the same sorts of things we'd already done with the counter pattern, but made it simpler for us to do them. As you went through the puzzles, though, you started doing some interesting movements that we hadn't been able to do before.

✓ Assessment Opportunity ▲

As students describe the blocks that they would make, ensure that they are relating the specific code that a block would use to the higher-level concept of what it would do. For example, a "velocity" block would move a sprite across the screen, and it would include blocks that use the counter pattern on a position property.

Students should describe the advantages of having these blocks, such as not needing to re-write the code all the time, or making it easier to read what the program is doing.

- Describe one of those movements, and how you made it.
- Describe another block that you'd like to have.
 - What would you name it?
 - What would it do?
 - What code would it hide inside?
 - How would it help you?

Remarks

All of the movements that we did today are possible without the new blocks, but it would be very complicated to code them. One of the benefits of blocks like velocity is that when we don't have to worry about the details of simple movements and actions, we can use that extra brainpower to solve

worry about the details of simple movements and actions, we can use that extra brainpower to solve more complicated problems. As you build up your side scroller game, we'll keep looking at new blocks that make things simpler, so we can build more and more complicated games.