

Lesson 23: Collisions

45 minutes

Overview

This lesson introduces collisions, another useful abstraction that will allow students to manipulate their sprites in entirely new ways. After a brief review of how they used the **isTouching** block, students brainstorm other ways that two sprites could interact. They then use **isTouching** to make one sprite push another across the screen before practicing with the four collision blocks (**collide** , **displace** , **bounce** , and **bounceOff**). This is the last time they will learn a new sprite behavior, and following this lesson students will transition to focusing on how they organize their increasingly complex code.

Question of the Day: How can programmers build on abstractions to create further abstractions?

Standards

Full Course Alignment

CSTA K-12 Computer Science Standards (2017)

- **AP** - Algorithms & Programming

Agenda

Warm Up (5 minutes)

Activity (35 minutes)

Wrap Up (5 minutes)

Objectives

Students will be able to:

- Describe how abstractions can be built upon to develop even further abstractions
- Model different types of interactions between sprites.

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

- **Collisions** - Slides ▼ Make a Copy

Vocabulary

- **Abstraction** - a simplified representation of something more complex.
Abstractions allow you to hide details to help you manage complexity, focus on relevant concepts, and reason about problems at a higher level.

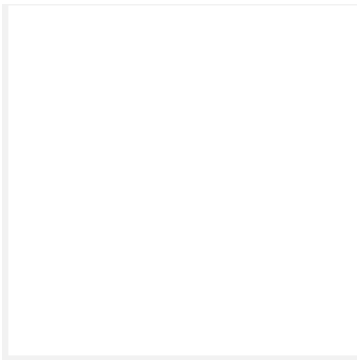
Introduced Code

- `setCollider(type, xOffset, yOffset, width/radius, height,`
- `sprite.bounce(target)`
- `sprite.bounceOff(target)`
- `sprite.bounciness`
- `sprite.collide(target)`
- `sprite.displace(target)`

Teaching Guide

Warm Up (5 minutes)

Display: Display the animated image. It is also available as a level in code studio.



Prompt: Using the blocks we already know how to use, how could we create the sprite interaction we can see in this program?



Code Prediction

Share: Allow students to share out their ideas.

Discussion Goal: The goal of this discussion is for students to brainstorm ways to solve the problem of having one sprite push another across the screen. There's no need for students to come to a consensus because they will each have a chance to try out a solution in the next level in Code Studio. Students should understand that it is possible to use blocks to produce the desired movement just with the blocks that they have already learned.

Remarks

The first part of the problem is figuring out when the two sprites are touching, but we already figured out how to do that and can now use the **isTouching** block. That means we don't need to think about those details anymore. Using abstraction to hide the complicated details in that part of the problem means we can focus on the new part.

Teaching Tip

Students have seen this vocabulary before, but given its importance to the chapter, it is introduced again here.

Vocabulary Review:

- **abstraction** - a simplified representation of something more complex

Question of the Day: How can programmers build on abstractions to create further abstractions?

Activity (35 minutes)

Transition: Move students onto Code Studio - level 2 where they will code the idea they made during the warm-up to make the giraffe sprite push the monkey sprite across the screen. When students finish level 2, have them try level 3 where they will make an elephant sprite push a hippo sprite down the screen.

Prompt: These were challenging problems, but we were able to solve them. What helped us to solve these problems?

Assessment Opportunity

Make sure students talk about the importance of higher-level blocks, such as **isTouching**, and that while these blocks don't provide new functionality, hiding the complexity of the code inside of a single block allows them to tackle more complex problems.

This is also a good time to call out how far the students have progressed in their skills since the beginning of the unit. This problem would have seemed almost impossible at the beginning of the year. Some things that made the problem easier to solve were:

- Preparation: The students brainstormed and thought about solutions before trying out their code.
- Cooperation: Students worked as a group to come up with a solution
- Abstraction: Students were able to use the **isTouching** and **velocityY** blocks to hide part of the solution's complexity.

Remarks

All of these things are very important, and they come up in Computer Science a lot. One thing that was particularly helpful was the **isTouching** block, which hid the complicated code that tells us whether the two sprites are touching. There's also a **displace** block that hides the code we just wrote, and some other blocks that hide the code for some other types of sprite interactions. You'll have a chance to try out these blocks in the next few levels.

Transition: Have students continue working on the remaining levels.

 2-5

Skill Building

2

3

4

5

 Teaching Tip

Guide to Programming Levels: Additional guidance for programming levels is provided in the [CSD Guide to Programming Levels](#). This document includes strategies and best-practices for facilitating programming levels with students.

 6

Practice

 7



Assessment

 Assessment Opportunity

Formative Assessment: This level can be used as a formative assessment. A rubric is provided in the level, and written feedback can be given to students. [Click here to learn more about giving feedback to students.](#)

 8

Challenges

Wrap Up (5 minutes)

Question of the Day: How can programmers build on abstractions to create further abstractions?

Vocabulary Review:

- **abstraction** - a simplified representation of something more complex

Prompt: How did having the **isTouching** block and the **velocityX** block make it easier to solve the problem of one sprite pushing another?

 Assessment Opportunity

Students should understand that these two blocks represent partial solutions to the problem of one sprite pushing the other, and that by hiding the details of those partial solutions, they can more easily focus on how to fit those partial solutions together to solve larger and more complex problems.



This work is available under a [Creative Commons License \(CC BY-NC-SA 4.0\)](#).

If you are interested in licensing Code.org materials for commercial purposes [contact us](#).