Lesson 16: Events in App Lab

45 minutes

Overview

In this lesson, students learn how to add screens to their apps, how to import screens created by other students, and how to program events to navigate between screens. Students learn basic event-driven programming by building up the model app that they started in the previous lesson. At the end of the lesson, students make a plan for how they will stay organized when importing each other's screens in tomorrow's lesson.

Question of the Day: How can I use events to create an app?

Standards

Full Course Alignment

CSTA K-12 Computer Science Standards (2017)

- ▶ **AP** - Algorithms & Programming
- ▶ **IC** - Impacts of Computing

Agenda

Warm Up (5 minutes)
- Getting Prepared
- Journal

Activity (35 minutes)
- Events in App Lab
- Preparing for Screen Import

Wrap Up (5 minutes)
- Journal

Objectives

Students will be able to:

- Create an event that detects and responds to user input

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

- **Events in App Lab** - Slides
  ▼ Make a Copy

For the students

- **Events in App Lab** - Video

Introduced Code

- `console.log(message)`
- `onEvent(id, type, function(event)){ ... }`
- `setScreen(screenId)`

Teaching Guide

Warm Up (5 minutes)

Getting Prepared

Distribute: Make sure each team has their materials available.

Journal

Prompt: Think back to when you were programming in Game Lab. When you wanted a game to respond to user interaction, how did you do it?

Discuss: Students should journal individually, then share in pairs before asking a few students to share with the class.

Discussion Goal: Students should remember that Game Lab uses a 'draw' loop that runs constantly. If they wanted to check for user interaction, they needed to use if-statements inside the loop to constantly check what the user was doing.

Alternate Prompt: If your students haven't completed Game Lab, you can adjust the prompt to have them think of a game they've played and ask how it gets input from the user. They may brainstorm things like pressing a button on a keyboard or clicking a character on a screen, or they could consider phone interactions like tapping the screen or swiping or pinching for an action to happen.

Remarks

> In Game Lab, we used the 'draw' loop to constantly check if a user had interacted with the game. This technique works really well in games and animations where you need to be updating a lot of things all the time. However, most phone apps are *not* doing things constantly. In fact, a lot of apps do absolutely *nothing* but wait for the user to click on something - this is called an **event**. Today, we're going to learn how to add events in App Lab by continuing to create the Recycle Helper app. Tomorrow, we'll add events to our own digital prototypes

Question of the Day: How can I use events to create an app?

Activity (35 minutes)

Video: Show students the **Events in App Lab** video in the slides.

Teaching Tip

To encourage active engagement and reflection, use one or more of the strategies discussed in the **Guide to Curriculum Videos**.

Events in App Lab

Code Studio: Have students log into their computers and open Code Studio. If you used Pair Programming in previous lessons, consider having students use it throughout these levels.

1-4 **Adding Screens**

1 2 3 4

Teaching Tip

Normalizing Mistakes and Supporting Debugging: As programming levels become more complex, students may find themselves with bugs in their code that they need to untangle. If this happens frequently, this can be a demoralizing experience for students and can affect their self-perception of how capable they are in class.

To counter this, we recommend normalizing bugs and mistakes as something that happens to everyone - it's just part of the process. You can show students our **Debugging Video**, which includes several students normalizing mistakes and discussing debugging strategies that students can use. Additionally, consider displaying the **Student Guide to Debugging** for students to reference throughout the unit and having **Bug Report Quarter-Sheets** available for students to use.

5-7 **Events and Linking Screens**

5 6 7

Assessment Opportunity

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**.

Circulate: Monitor students as they complete the levels in Code Studio. As they progress through the bubbles, they should be adding to the Recycle Finder app from previous lessons. Students will spend most of the class completing these levels. As teams start to finish up, begin to transition to the next stage of this activity

Remarks

> A key part of adding events to your app was having different screens to link to. Today we learned one way to do that is by importing other screens. We will be doing this same process with your teammates - importing their screens into your app. To prepare for this, we need to plan how we will share our projects with each other.

Preparing for Screen Import

Do This: Have students discuss as a group how they will share their screen's import URLs with each other. They should decide on a plan that they will implement tomorrow.

Teaching Tip

Preparing to Share: Having a system for students to share their project URLs is important for tomorrow's lesson. Common strategies include having a shared Google Doc where students paste their URLs, or send emails to everyone in the group and creating an email thread, or using your school's learning management system to have students communicate with each other. You can let students decide which method they would like to use, or you can instruct students to use a specific strategy to simplify the process.

Wrap Up (5 minutes)

Collect: Collect all the materials from each team in a safe storage location.

Journal

Prompt: What do you think will be a challenge when it's time to import screens and continue creating your digital prototypes tomorrow? How can your teammates help with these challenges?

Discuss: Have students journal individually first, then have students share with a partner before asking a few students to share with the class.

Discussion Goal: Students may predict that they may have trouble staying organized with all the different IDs, or they may make coding mistakes with the events, or they may have import errors when trying to import screens. Encourage students to consider how their teammates can be a resource, such as by acting like a "thought partner" to help with debugging or by using their Screen Design Activity Guide to help stay organized with IDs.