Lesson 13: Customizing Apps

45 minutes

Overview

In this lesson, students will explore how to customize the code of their app to make additional changes to the design of their app. They will start by exploring a single-screen app and then practice expanding the app to two-screens and updating the code to use the new design mode elements. After this, students help create a Driver Alert app that requires changes to the code using new design mode elements. Using the skills from this lesson, students will be able to create multi-screen apps where questions can appear on multiple screens instead of a single screen.

Question of the Day: How can I customize the code for a machine learning app?

Standards

Full Course Alignment

CSTA K-12 Computer Science Standards (2017)

▶ **AP** - Algorithms & Programming

Agenda

Warm Up (5 minutes)
Journal

Activity (35 minutes)
Driver Alert System

Wrap Up (5 minutes)
Journal

Objectives

Students will be able to:

- Update the default model code to use new inputs in a machine learning app
- Use the Model Card to create new input elements for a machine learning app

Preparation

- Review the Code Studio levels before the lesson to be familiar with changing the input element
- Check the **"Teacher's Lounge"** forum for verified teachers to find additional strategies or resources shared by fellow teachers

Links

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

For the teachers

- **Customizing Apps** - Slides
  ▾ Make a Copy

Teaching Guide

Warm Up (5 minutes)

Journal

Display: Show students the example app, Raspado Recommender

Prompt: *What changes would you make to this app to improve how easy it is to use?*

Have students journal individually, then discuss with a neighbor before regrouping

Discussion Goal: Students can pull from previous lessons to make suggestions on how to improve this app: add a theme to improve the design, create a home-screen that the user sees first, add images or other visuals, etc. In addition to these suggestions, guide students to consider that there are too many options on one screen and that the app should be multiple screens instead. This will segue into today's lesson, where we look at additional ways to customize and improve our apps.

Remarks

> There comes a point where our AI apps have too many features for a single screen. In these cases, it can be useful to split up our app into multiple screens and spread out the questions that we ask users. This makes the app easier to use, and makes it easier to group questions together. Today, we'll learn how to split up our apps into multiple screens and how to update the code in our app to match the new screens.

Question of the Day: How can I customize the code for a machine learning app?

Activity (35 minutes)

Driver Alert System

Code Studio: Have students log into Code Studio Level 1. In this app, they see a similar Raspado Recommender as in the warm up - however, the questions have been split across 2 screens instead of just 1.

Circulate: Monitor students as they follow the instructions on this screen. They can click on the images within the instructions of the level for hints. This is the first time students are editing the pre-generated code for models, so be sure to check in with students before continuing to ensure they understand how to complete this level.

1 **Raspado Recommender**

Teaching Tip

Design Elements Off the Screen: This level prepares students for a situation they may encounter in their chapter project: having so many features that the design elements appear off the screen. This tends to happen when using more than 6 features in a model. If students encounter this in later levels, referring them back to this level can be useful to remind students how to problem-solve and rearrange the elements on the s creen.

Remarks

> Now that we know how to edit our code to use new design mode elements, let's take a look at a more advanced example. In the next series of levels, we're going to help build a Driver Alert System that can be used by cars to help warn drivers of dangerous conditions.

Code Studio: Have students continue to the next level in Code Studio. In these next set of levels, students will customize an app so each question is on a separate screen, which allows you to add more information for the user. As students complete each level, they will learn how to update their code to use custom design elements rather than the ones that were generated by default.

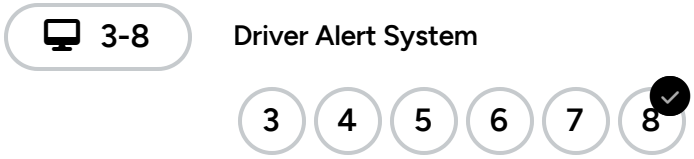
2 **Driver Alert System**

Discuss:

- *How many features does this app have? What are they?*
- *What is the label for this app? What are the possible outputs?*

Discussion Goal: Students should realize there are 4 features - number of cars, weather, temperature, and time of day. As students respond, ask them to identify whether the features are categorical or numerical, and how do they know? Students may respond based on the design elements on the screen (dropdowns for categorical, text boxes for numerical), or they may have looked at the Model Card for this information. They should also identify three possible label values - danger, warning, and normal. Point out that this information is easier to see in the model card rather than testing the app over and over again.

Code Studio: Have students continue through the next several levels in Code Studio, where they will recreate this app.



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**.

Circulate: Monitor students as they recreate this app. Students won't be able to test the app until it is fully complete, which may be a challenge for some students. Here are some things to be on the lookout for as students work:

- **Level 3:** Students practice deleting the default elements that are generated when you first import a model. This is a step that will be important when students create their own custom apps in later projects.
- **Level 4 and 5:** Students are only updating the code of their app to use the new design elements for the Number of Cars and Weather features. They can use the hints within the level instructions for clues on what to do in each step.
- **Level 6:** Students update both the design and the code of their app to collect data for the Temperature feature. Help students keep track of the new text input element they add to the temperature screen, and make sure the ID of the element matches the one they use in the `getText` block of their code.
- **Level 7:** Students update both the design and the code of their app to collect data for the Time of Day feature. Adding the design element and updating the code is similar to the last level, but this step has the added requirement of using the Model Card to check the possible values are for this categorical feature. Students may make spelling or capitalization errors that can effect their code - encourage students to be very careful to match their dropdown options exactly with the model card.
- **Level 8:** Students have a chance to run their app and verify that it works. Students will only need to make changes to the code on this level if their program doesn't work. Otherwise, if the app works correctly, students can continue to the next level.

Choice Levels: Once students have created their Driver Alert System App, they can choose from several example activities to continue improving their app. Each level uses the same Driver Alert app they've already built as a template to add to. Students can choose to complete as many of these levels as they would like - each one lets them continue to customize and improve their app.

Teaching Tip

Repeated Levels: The choices in this level are identical to the choices in the last lesson. This is intentional, so students have multiple opportunities to learn the new design tricks or code blocks, or continue to practice the new blocks they learned last lesson. Encourage students to choose a new challenge for today's app that's different from the ones they chose yesterday.

9 **You Choose: Improve Your App**

Wrap Up (5 minutes)

Journal

Prompt: *What is one benefit to having features spread across multiple screens in an app? What is one drawback to having features on multiple screens in an app?*

Discussion Goal: Student answers may vary, but some examples are:

- Students may realize that having features on multiple screens can make it easier to describe the feature and easier for the user to focus on a single question.
- Students may also realize that this increases how "long" the app is, and it adds additional work to update the code with the new elements.