

# Unit 3: Intro to App Design

**Lesson 1:** [Introduction to Apps](#)

**Lesson 2:** [Introduction to Design Mode](#)

**Lesson 3:** [Project - Designing an App Part 1](#)

**Lesson 4:** [Project - Designing an App Part 2](#)

**Lesson 5:** [The Need for Programming Languages](#)

**Lesson 6:** [Intro to Programming](#)

**Lesson 7:** [Debugging](#)

**Lesson 8:** [Project - Designing an App Part 3](#)

**Lesson 9:** [Project - Designing an App Part 4](#)

**Lesson 10:** [Project - Designing an App Part 5](#)

**Lesson 11:** [Assessment Day](#)

# **Unit 3 - Lesson 1**

## **Introduction to Apps**

# Warm Up

●○○

## Prompt:

What are apps? How do we interact with them? What kind of things do apps do?

# Activity



# App Exploration

## Water Conservation Tips

It's important that we all do our part to use less water. Click through this app for tips for conservation ideas.

[Spanish](#)[English](#)[Next](#)

## Bird Quiz!

Think you know birds? Time to test your knowledge.

[Begin](#)

Here in beautiful, sunny Hamilton our residents are working hard to add beauty to our city streets. The current plan approved by the town committee involves placing baskets of flowers on all street lamp posts located on Main and Cross streets from May through October.



Red



Blue



Purple



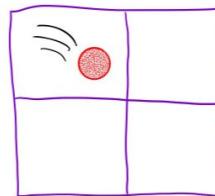
Click to see the different color options for the flowers

Hamilton Township Improvement Project

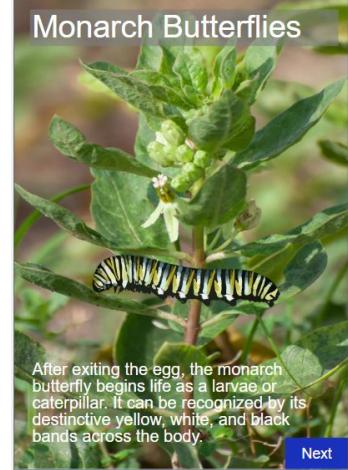


## 4-Square = 4 Times the Fun

Have you wanted to learn how to play the fun, social game of 4-square, but didn't know where to start? This is the app for you!

[More Info](#)

## Monarch Butterflies



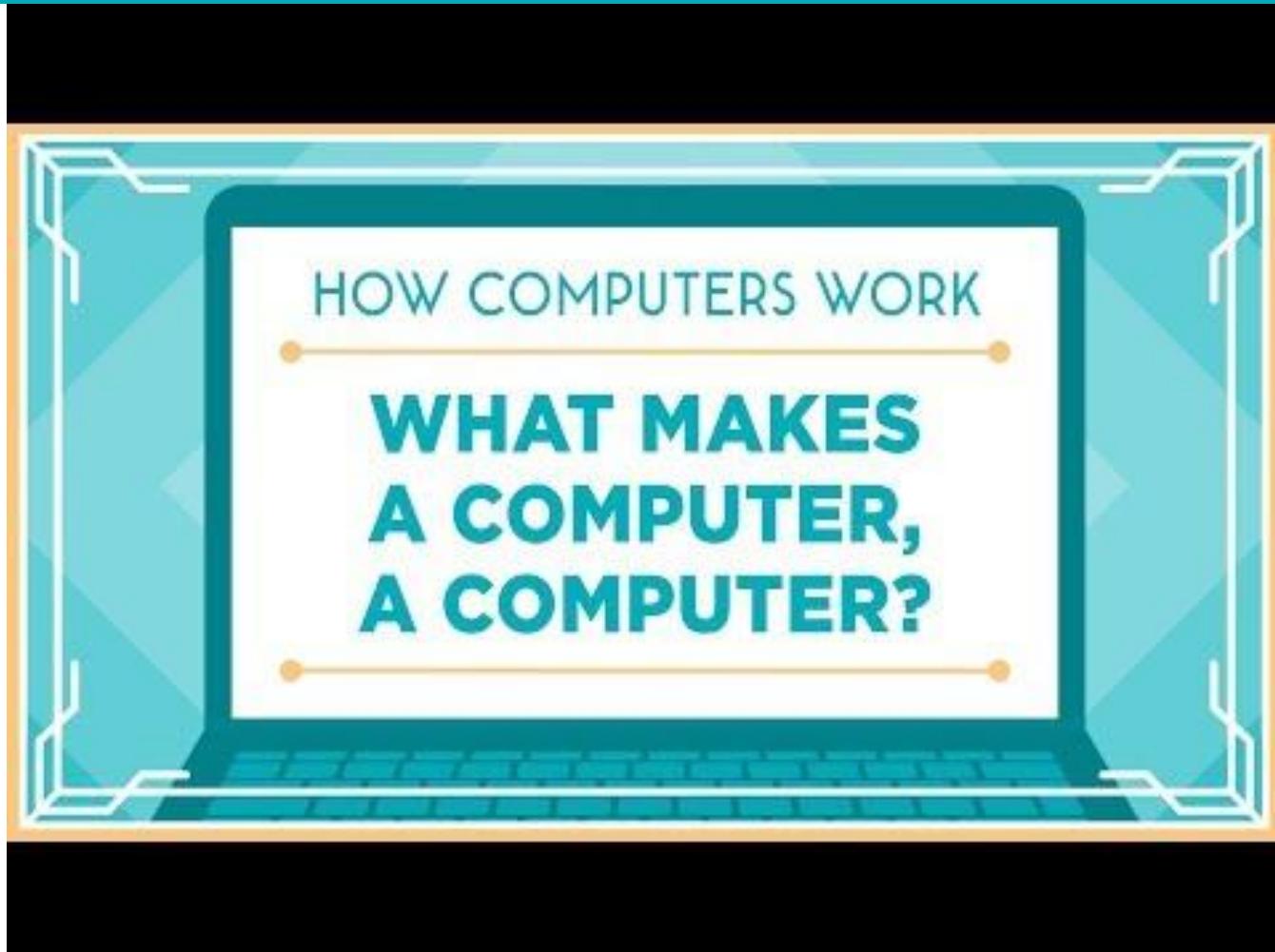
After exiting the egg, the monarch butterfly begins life as a larvae or caterpillar. It can be recognized by its distinctive yellow, white, and black bands across the body.

[Next](#)

## Prompt:

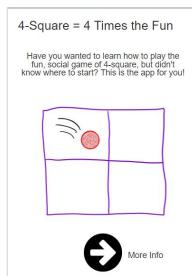
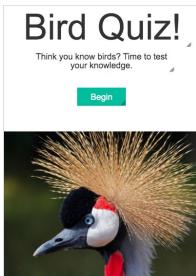
With a partner discuss the following and note down in your journal:

- How does the user interact with the app?
- What is the overall purpose of the app?
- Who is the target audience?



# App Investigation

For this part, start at Level 6



## Do This:

With your partner, take another look at the sample apps you explored before by navigating to the App Investigation starting at Level 6. Consider what the inputs and outputs are for the apps.

Note these down in your journal.

# Wrap Up





## Prompt:

Think of your favorite app. Discuss with a partner what the user interface looks like and the inputs and outputs.



**User Interface:** the inputs and outputs that allow a user to interact with a piece of software. User interfaces can include a variety of forms such as buttons, menus, images, text, and graphics.



**Input:** data that are sent to a computer for processing by a program. Can come in a variety of forms, such as tactile interaction, audio, visuals, or text.



**Output:** any data that are sent from a program to a device. Can come in a variety of forms, such as tactile interaction, audio, visuals, or text.

# **Unit 3 - Lesson 2**

## **Introduction to Design Mode**

# Warm Up

●○○

## Prompt:



What is a common app that you use?  
Take a minute to sketch the User Interface of the main screen. Note how the user interacts with the app.

# Activity



# Introduction to Design Mode

The screenshot displays a mobile application design interface. On the left, there is a navigation bar with a teal background and white text. It features a circular icon with the number '1' and a horizontal row of seven small circles.

The main area shows a screen titled "homeScreen". The content includes a text block about a town improvement project, a large red square, a flower basket, and three colored buttons labeled "Red", "Blue", and "Purple". A callout text says "Click to see the different color options for the flowers". Below this is the title "Hamilton Township Improvement Project" and a "Run" button.

To the right, there is a "Design Toolbox" with various UI element icons: Button, Text Input, Label, Dropdown, Radio Button, Checkbox, Image, Canvas, Screen, Text Area, Chart, and Slider. A "Properties" panel on the far right shows the current element's properties: id (set to "homeScreen"), theme (Classic), background color (#ffffff), and image (empty).

**Instructions**

**Do This**

- Look at the elements on the screen. Try to move some of them around.
- What properties can you change? Is there anything you can't change?

# Wrap Up





## Prompt:

- What elements collect input?
- What elements display output?
- Do you think there are elements that can do both?

# **Unit 3 - Lesson 3**

# **Project - Designing an App Part 1**

# Warm Up

●○○

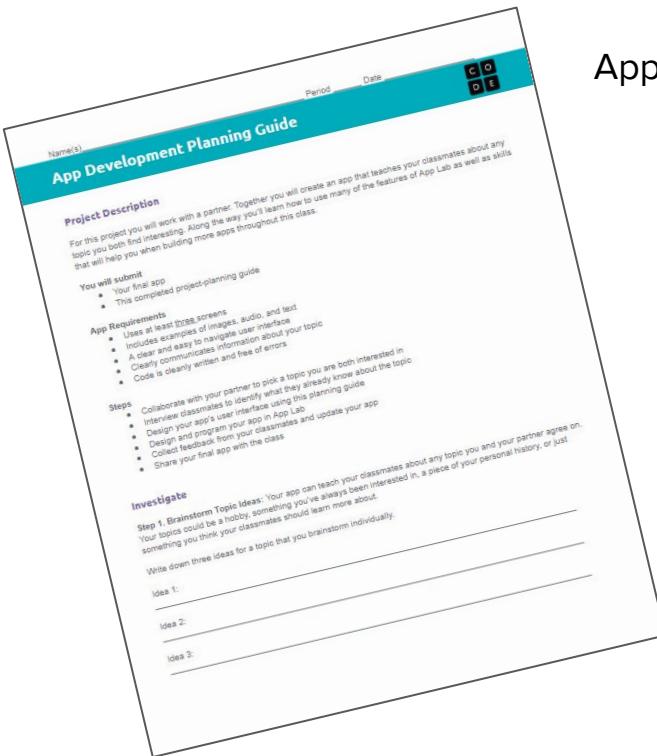
**Prompt:** People design user interfaces to meet a user's needs, but they don't always get it right.

- Have you ever used an app where the user interface didn't actually meet your needs?
- What was the problem?
- What do you think the designers didn't understand about you or your needs?

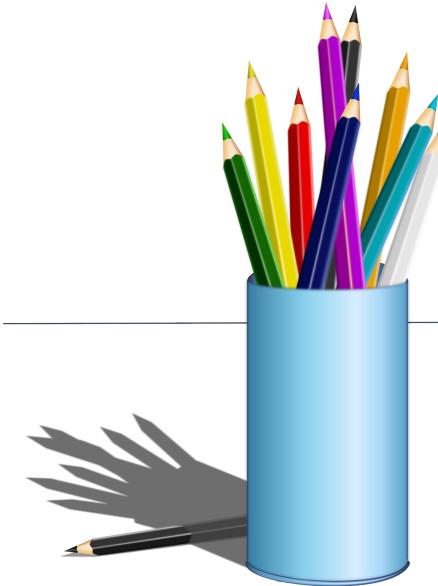
# Activity



# Designing an App Part 1



**You should have:**  
App Development Planning Guide





Name(s): \_\_\_\_\_ Date: \_\_\_\_\_

Period: \_\_\_\_\_ CO DE

### App Development Planning Guide

**Project Description**

For this project you will work with a partner. Together you will create an app that teaches your classmates about any topic you both find interesting. Along the way you'll learn how to use many of the features of App Lab as well as skills that will help you when building more apps throughout this class.

You will submit:

- Your final app
- This completed project-planning guide

**App Requirements**

- Uses at least 10 screens.
- Includes examples of images, audio, and text.
- A clear and easy to navigate user interface.
- Clearly communicates information about your topic.
- Code is clearly written and free of errors.

**Steps**

- Collaborate with your partner to pick a topic you are both interested in.
- Interview classroom to identify what they already know about the topic.
- Design your app's user interface using this planning guide.
- Design and program your app in App Lab.
- Collect feedback from your classmates and update your app.
- Share your final app with the class.

**Investigate**

Step 1. Brainstorm Topic Ideas: Your app can teach your classmates about any topic you and your partner agree on. Your topics could be a hobby, something you've always been interested in, a piece of your personal history, or just something you think your classmates should learn more about.

Write down three ideas for a topic that you brainstorm individually.

Idea 1: \_\_\_\_\_

Idea 2: \_\_\_\_\_

Idea 3: \_\_\_\_\_

# Step 1:

# Brainstorm Topic Ideas

**Do This:**  
Choose a partner!

## **Tip:**

Keep an eye out for bias!  
Collaboration with others is key.

**Step 2. Choose One Topic:** Now talk through your ideas with your partner. Together pick a topic both of you are interested in teaching your classmates about. Explain in a few sentences what would be covered. For example, if your topic is Basketball, you would write a few sentences explaining that you would cover the rules and the origin of the sport.

Our Topic:

---

---

---

---

---

## **Step 2:**

# Choose One Topic



**Step 3. Survey Your Classmates:** To design your app you'll need to understand your users. For this project your user is your classmates, and you'll need to understand what they already know about your topic.

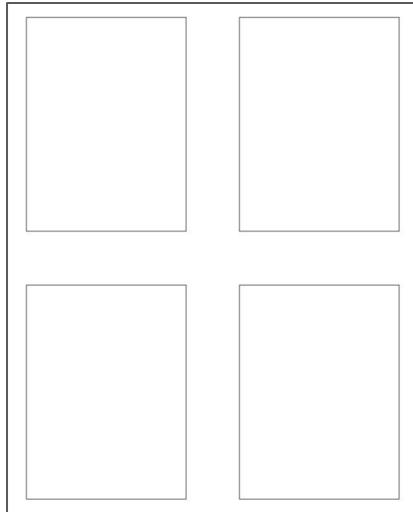
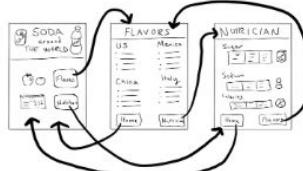
Find two classmates and talk to them about your topic for a couple minutes. Then fill in this table.

Name	What do they already know about your topic?	What do they need or want to learn about your topic?

## Step 3: Survey Your Classmates

**Design**

**Step 4. Design the User Interface:** In the space on the following page, draw a rough sketch of your user interface. This means you should include all the buttons, text, and images that the user will be able to use. Write notes or draw arrows showing how different user interface elements should work. For example, if clicking a button takes me to another screen, I should draw an arrow from that button to the drawing of the screen.



# Step 4:

## Design the User Interface

# Wrap Up





## **Prompt:**

How did talking with the users of your app impact your design decisions?

# **Unit 3 - Lesson 4**

# **Project - Designing an App Part 2**

# Warm Up

●○○

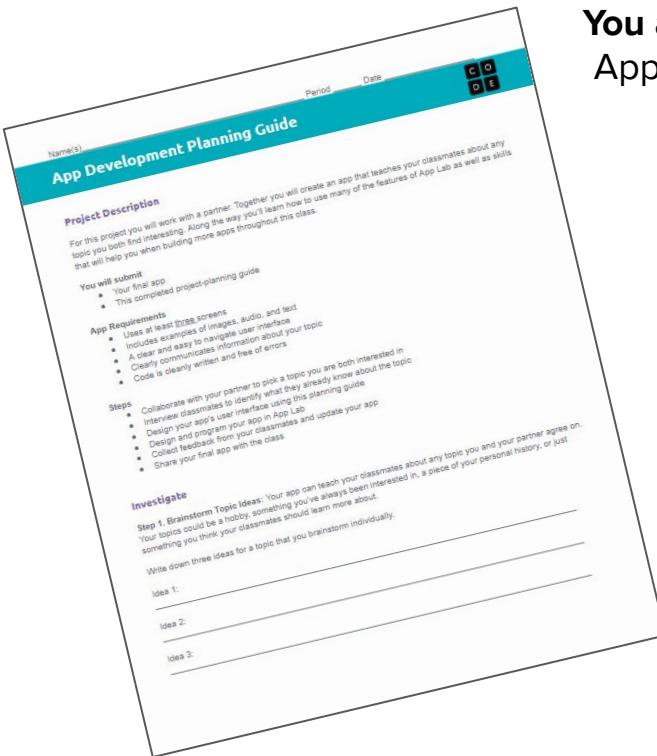
## **Prompt:**

Why is it important to plan out the design of an app?

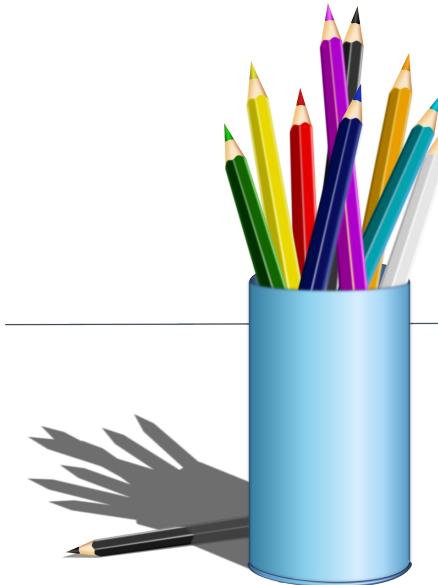
# Activity



# Designing an App Part 2

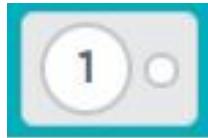


**You and your partner should have:**  
App Development Planning Guide  
Pen/Pencil



# Step 5: Create Your User Interface

**Do This:** Navigate to Code Studio, Lesson 4, Level 1. Follow the instructions.



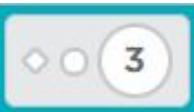
The screenshot shows a mobile application prototype titled "Bird Quiz!". The main screen features a large image of a bird's head and a "Begin" button. Below it is a "Run" button. To the right, there is an "instructions" section with the heading "Do This" and two bullet points:

- Take a look at the prototype below for the Bird Quiz App.
- Run the app, and think about how the prototype helped in the development of this app.

Below the instructions is a flowchart diagram illustrating the quiz logic. It starts with a "Bird Quiz" box containing a "Begin" button. An arrow labeled "Right" leads to a box asking "What bird is fast on land?", which lists "Ostrich" and "Emu". From "Emu", an arrow labeled "Wrong" leads to a box asking "What bird can live longest?", listing "Swan" and "Budgie". From "Budgie", an arrow labeled "Wrong" leads to a box asking "What bird can fly?", listing "Parrot" and "Eagle". From "Eagle", an arrow labeled "Right" loops back to the initial "Bird Quiz" box. There are also "Wrong" arrows from "Ostrich" to the "longest" box and from "Swan" to the final "fly" box.

# Do This: Navigate to Level 2.

## Start building your user interfaces!



The screenshot shows a mobile application design interface. On the left, there's a preview window titled "screen1" with a "Run" button at the bottom. In the center, a "Design Toolbox" panel lists various UI components: Button, Text Input, Dropdown, Label, Radio Button, Checkbox, Image, Canvas, Screen, Text Area, Chart, and Slider. To the right, a "Properties" panel is open for a "Button" component, showing settings for "id" (set to "screen1"), "theme" (set to "Default"), "background color" (#ffffff), and "image". The top right corner of the screen has a "Instructions" section with the heading "Do This" and two bullet points: "Look at the user interface design in your planning guide" and "Recreate the screens here".

If you divided the screens with a partner, here's how to combine them into one project.



Partner A



Partner B

1. Choose one partner to host the project (**Partner A**).
2. **Partner A** clicks the screen dropdown, then clicks "Import screen"
3. Paste in the share link from **Partner B**.
4. **Partner A** select to import all of the screens and assets.
5. **Partner A** set the home screen to be the default screen (Hint: Go to design mode and click on the screen)

The screenshot shows the 'Import screens' step in the Code.org import interface. The 'Design' tab is selected. A dropdown menu is open with options: 'homeScreen', 'homeScreen', 'Import screen...', and 'New screen...'. The 'Import screen...' option is highlighted with a blue selection bar. To the right, there is a modal window titled 'Import screens' with instructions to copy a share link and a URL input field containing 'https://studio.code.org/projects/applab/3GSYZSsCri4HeCDgBeOoJRpjtjQ'. A 'Next' button is at the bottom right of the modal. Below the modal, another window titled 'Import from Project: Hamilton Township' lists imported assets: 'Screens' (homeScreen, trashScreen, gardenScreen) and 'Other Assets' (Blue.mp3). An 'Import' button is at the bottom right of this window.

# Wrap Up





## **Prompt:**

Were there any changes you had to make to your original design once you transferred it to the screen?

# **Unit 3 - Lesson 5**

## **The Need for Programming Languages**

# Warm Up

●○○

## Prompt:

Write down three different reasons you would call a set of instructions "bad".

Be ready to share with a neighbor.

# Activity

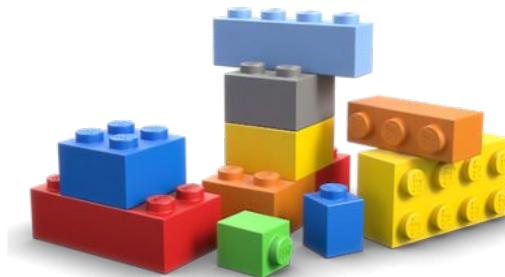


# You and your partner should have:

Pen / Pencil

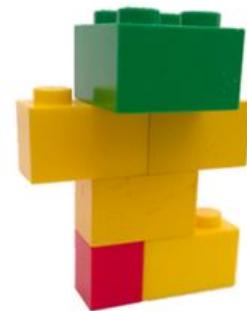
Sheet of paper

A small set of Legos or other blocks



## Step 1: Design

- Put 5-6 pieces together
- All pieces must be connected



## Step 2: Record

- Take a photo or draw a picture
- Color matters



## Step 3: Write instructions

- Write instructions for building your design
- Be as clear and precise as possible
- Just words, no drawings, diagrams, or pictures



## **Step 4: Trade**

- Take apart your design, then trade pieces and instructions with another group

## **Step 5: Build**

- Try to build the design following the instructions

## **Step 6: Compare**

- Compare your design to the picture the other team recorded

## **Step 7: Repeat**

- If you have time try this activity with one or two other groups' instructions.

# Wrap Up





## Prompt:

When you or your classmates made mistakes following instructions today what "went wrong"? Try to be as specific as possible.

**Prompt:** Imagine we were going to redesign human language to be really good for giving clear instructions. What types of changes would we need to make?

# Unit 3 - Lesson 6

## Intro to Programming

# Warm Up

●○○

# Activity



# Intro to Programs

## You and your partner should:

- Work together to complete the “Do This”
- Talk through the “Discuss” prompts together
- “Modify” the code to follow the directions given
- Be prepared to share your discussions with the class

The image shows a Scratch workspace titled "Lesson 6: Programs Investigate". The workspace includes a stage with a script area containing a script for a cat sprite. The script consists of a "say [Hi! v1]" block followed by a "say [This is a very simple program. v1]" block. Below the stage is a "Run" button. To the right of the stage is a "Coder" interface with tabs for "Instructions", "Do This", "Discuss", and "Modify". The "Do This" tab contains the instruction "Run this program to see what it does?". The "Discuss" tab contains the question "Why do you think some information is in quotes and some is not?". The "Modify" tab contains the instruction "Add two lines of code, one that displays a string and one that displays a number". At the bottom is a code editor with a "Toolbox" tab, a "Variables" tab (selected), a "Workspace" tab, and a "Version History" tab. The code editor shows the following script:

```
1 console.log("Hi!");
2 console.log("This is a very simple program.");
3 console.log("You can display text in quotes");
4 console.log("But you can also display numbers, like this");
5 console.log(100);
6 console.log("Try adding some code of your own below!");
```

# Wrap Up





## Prompt:

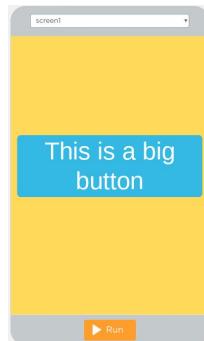
Think about your experiences today and in the previous lesson. How is a programming language different from natural language?

**Program Statement:** a command or instruction. Sometimes also referred to as a code statement.

```
setProperty(▼"bigButton", ▼"text", ▼"Click me");
```

```
console.log("Hi!");
```

**Program:** a collection of program statements. Programs run (or “execute”) one command at a time.



```
1 console.log("Starting my program!");
2 setProperty(▼"bigButton", ▼"background-color", ▼"blue");
3
```



**Before**

**Command Runs**  
(yellow outline while running)

**After**



Two different ways for programs to run

**Sequential Programming:** program statements run in order, from top to bottom.

- No user interaction
- Code runs the same way every time

The Scratch workspace shows a script for a blue button labeled "bigButton". The script consists of five sequential steps: 1. A log message "Starting my program!". 2. A setProperty step to change the background color of the button to blue. 3. A setProperty step to change the text of the button to "This button is blue!". 4. A setProperty step to change the height of the button to 150 pixels. 5. A log message "Ending my program!". The stage has a yellow background with a grey footer bar containing a "Run" button.

```
1 console.log("Starting my program!");
2 setProperty( ▯ "bigButton", ▯ "background-color", ▯ "blue");
3 setProperty( ▯ "bigButton", ▯ "text", ▯ "This button is blue!");
4 setProperty( ▯ "bigButton", ▯ "height", ▯ 150);
5 console.log("Ending my program!");
6
```

**Event Driven Programming:** some program statements run when triggered by an event, like a mouse click or a key press

- Programs run differently each time depending on user interactions

The Scratch workspace shows a script for a blue button labeled "bigButton". The script is triggered by a "click" event and contains four steps: 1. A log message "You clicked the button!". 2. A setProperty step to change the text of the button to "You clicked me!". 3. A setProperty step to change the background color of the button to blue. 4. A close bracket step to end the event loop. The stage has a yellow background with a grey footer bar containing a "Run" button.

```
1 onEvent( ▯ "bigButton", ▯ "click", function(){
2   console.log("You clicked the button!");
3   setProperty( ▯ "bigButton", ▯ "text", ▯ "You clicked me!");
4   setProperty( ▯ "bigButton", ▯ "background-color", ▯ "blue");
5 })
```

# Unit 3 - Lesson 7

## Debugging

# Warm Up

●○○

## Prompt:

Your friend calls and says "I can't get music to come out of my speakers"

Write a quick list of everything you'd ask them or have them check to try to fix the problem.

# Activity





## Debugging: the process of finding and fixing problems in code

### Describe

#### The Problem

What do you expect it to do?

What does it actually do?

Does it always happen?

### Hunt

#### For Bugs

Are there warnings or errors?

What did you change most recently?

Explain your code to someone else

Look for code related to the problem

### Try

#### Solutions

Make a small change

### Document

#### As You Go

What have you learned?

What strategies did you use?

What questions do you have?

# Wrap Up





## Prompt:

Share any debugging tips you recorded today with your neighbor.

Be ready to share with the class.

## Debugging Strategies

### Keep your code clean

- Use clear, meaningful IDs for your elements
- Keep your code organized in chunks that do the same thing
- Use comments to explain your code
- Write code using blocks

### Run your code

- Run your code a lot, every time you add a command or two
- Slow down your code with the speed slider. Watch how it runs closely
- Use console.log to get output. Add extra output statements throughout your code to get feedback on what parts are running.

### Use classmates and resources

- Talk out the problems with a partner or classmate
- Compare your code to examples that you know work
- Read documentation to know how a block is supposed to work
- Hand trace your code to track what's happening.



**Documentation:** a written description of how a command or piece of code works or was developed.



A screenshot of a documentation page for the 'playSound' command. The page includes a sidebar with categories like 'UI Controls', 'Variables', 'Data', 'Turtle', 'Sensing', and 'Math'. The main content area has the following sections: 'playSound' (Category: UI Controls), 'Description' (Play the MP3 sound file from the specified URL), 'Examples' (1 // Play a goal sound; playSound("https://studio.code.org/blockly/media/assets/studio/1\_goal.mp3");), and 'Comments' (Today's quick play sounds is made from more engaging. You can add sounds to your apps that are triggered by events on UI elements, or based on user movements, or not based on other app code. There are two ways to fill in the url string for the parameter: 1. Copy the URL of the file on the web. In most browsers you can simply right-click (or control-click on a Mac) on a sound file and you'll see a menu with a few options. One will be to copy the URL of the sound. Note: We have listed some exciting audio files that you can use in your app below. 2. Upload your own sound to App Lab. You can upload sound files saved on your computer to your app in App Lab. • Click the 'Choose' arrow in the 'image' URL field and then click 'Choose...' playSound("https://studio.code.org/blockly/media/assets/studio/1\_goal.mp3"); / choose... • Then click the 'Upload File' button near the top of the window. Choose Assets You can edit your file by clicking 'Open' to edit a new asset for the project. Choose file... • Then choose the file from your computer by navigating to it. • Once its uploaded click 'Choose' next to it. This will insert the name of the file into the URL field. Because you have uploaded it, it doesn't need to be an HTTP reference. Examples 1 // Play a goal sound; playSound("https://studio.code.org/blockly/media/assets/studio/1\_goal.mp3");).

**Comment:** form of program documentation written into the program to be read by people and which do not affect how a program runs.

```

1 // When the user clicks the cat button
2 // play a meow sound and show cat image and text
3 onEvent(catButton, click, function() {
4     setProperty(messageLabel, text, "Cats Rule!");
5     playSound(sound://category_animals/cat.mp3);
6     setProperty(petImage, image, https://cdn.pixabay.com
7 })
  
```

# **Unit 3 - Lesson 8**

## **Project - Designing an App Part 3**

# Warm Up

●○○



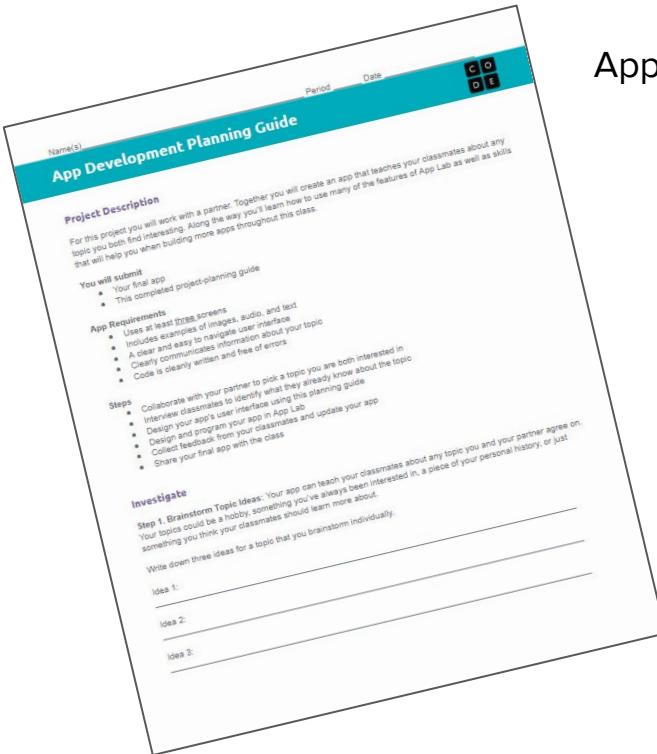
**Prompt:**  
What makes a good partner?

# Activity



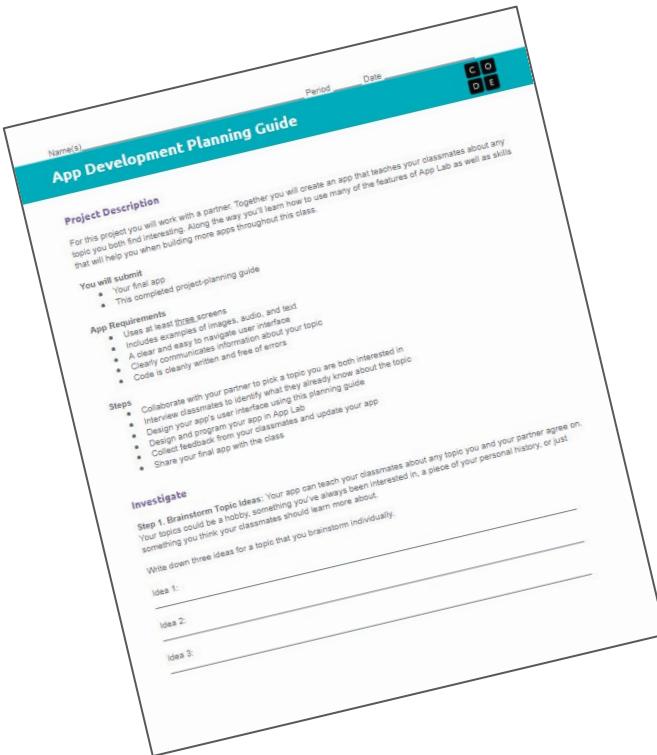
# Designing an App Part 3

You should have:  
App Development Planning Guide



# Step 5:

# Start Building Your App



Element ID	Action	What happens?
“dogButton”	“click”	<i>A picture of a dog appears The background of the screen changes to green</i>



OF  
COURSE  
IT'S  
NORMAL

# What is Pair Programming?



## Driver

Manipulates the keyboard  
and the mouse



## Navigator

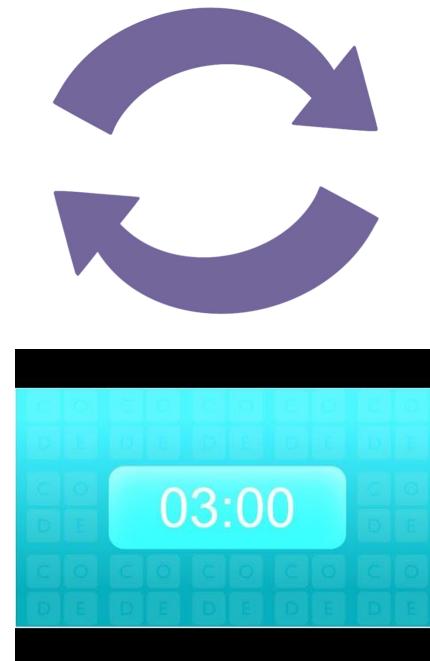
Keeps track of the big picture.  
Guides towards the goal.

You will swap between roles every 3 minutes

# Swap every three minutes



**Driver**



**Navigator**



# Wrap Up





## **Prompt:**

How does Pair Programming help  
when working on a project?

How does it help with the debugging  
process in particular?



**Pair Programming:** a collaborative programming style in which two programmers switch between the roles of writing code and tracking or planning high level progress

# **Unit 3 - Lesson 9**

# **Project - Designing an App Part 4**

# Warm Up

●○○

## Prompt:

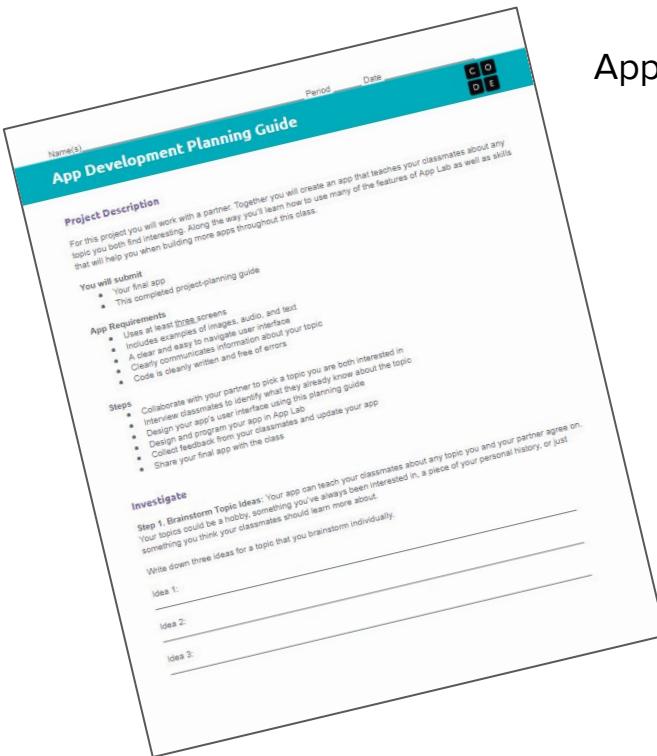
Think of times when you've received helpful feedback on school work, a hobby, or a sport.

What makes good feedback?  
What makes bad feedback?

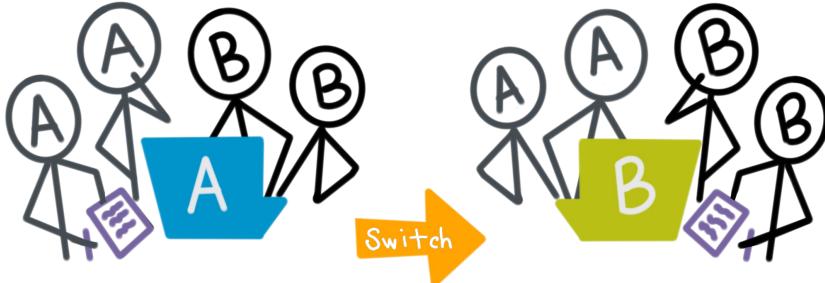
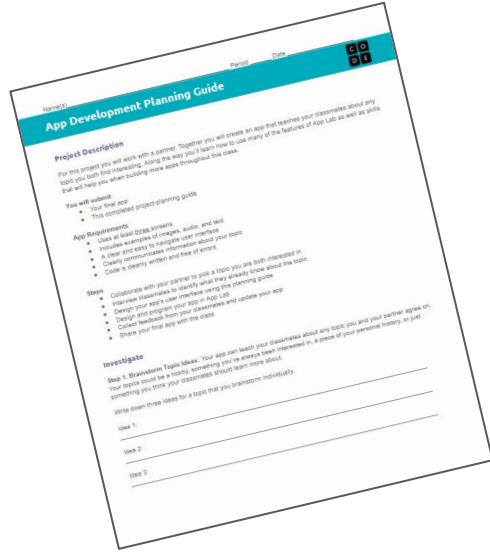
# Activity



# Designing an App Part 4



**You should have:**  
App Development Planning Guide



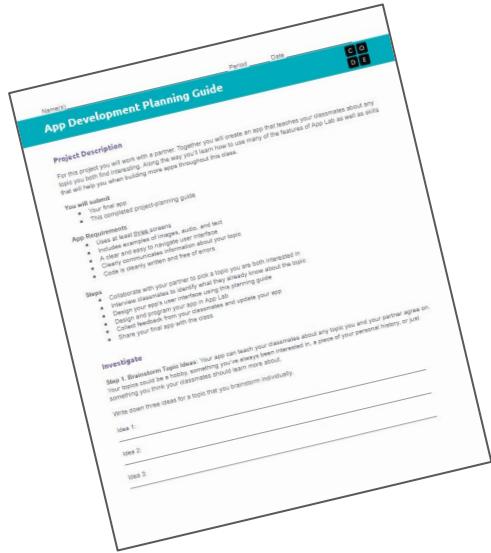
# Step 6:

## Testing and Feedback

Name	Things that could be improved based on watching them use the app	Improvements this person recommends

# Wrap Up





# Step 7:

## Pick Improvements

**Step 7. Pick Improvements:** Pick at least one improvement you plan to make to your app based on feedback you collected from your classmate.

Improvement 1:

---

---

Improvement 2 (Optional):

---

---



## **Prompt:**

Why is it important to get feedback from others while building your app?

What is the value of getting this feedback even if you aren't finished with your app?

# **Unit 3 - Lesson 10**

# **Project - Designing an App Part 5**

# Warm Up

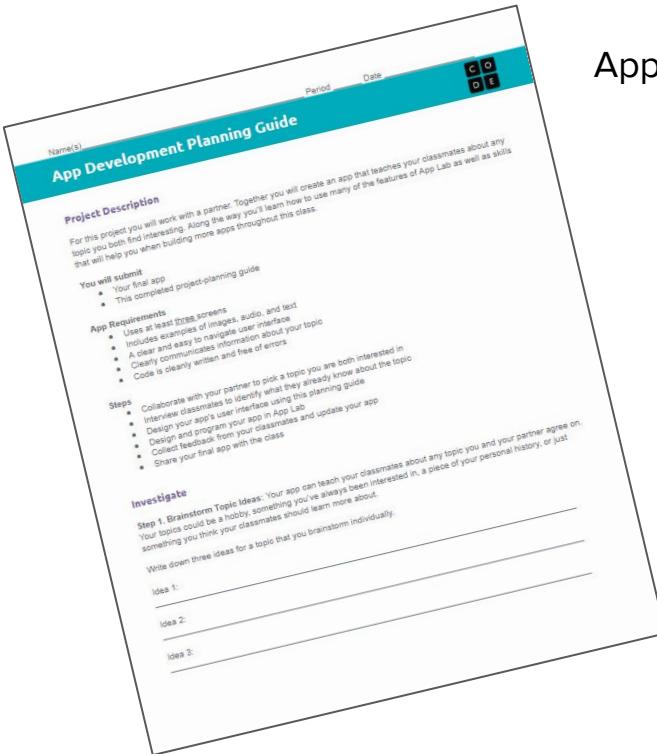
●○○

# Activity



# Finish Your App

You should have:  
App Development Planning Guide



Category	Extensive Evidence	Convincing Evidence	Limited Evidence	No Evidence
User Interface Screens	User interface includes at least three screens.	User interface includes two screens.	User interface is on a single screen.	The screen is blank.
User Interface Navigation	The user can easily navigate between all screens.	The user can easily navigate between most screens.	The user can easily navigate between some screens.	The user cannot navigate between screens.
User Interface Elements	A app includes at least one example of each element type: <ul style="list-style-type: none"><li>- Text</li><li>- Images</li><li>- Audio</li></ul>	The app includes at least one example of two or more of the following: <ul style="list-style-type: none"><li>- Text</li><li>- Images</li><li>- Audio</li><li>- Video</li></ul>	The app includes at least one example of one of the following: <ul style="list-style-type: none"><li>- Text</li><li>- Images</li><li>- Audio</li><li>- Video</li></ul>	The app includes no text, images, or audio.
Code	Code runs without errors.	Code runs with a few errors.	Code does not run or has a lot of errors.	Code is blank.
Element IDs	Screen elements all use meaningful IDs.	Screen elements use IDs, but they are not meaningful.	Some screen elements use meaningful IDs.	Screen elements do not use meaningful IDs.
App Topic	Topic is clearly communicated and explained.	Topic is communicated and explained.	Topic is communicated well.	App appears to be a random collection of elements with no clear topic.
App Development Planning Guide	Planning guide is fully completed.	Planning guide is mostly completed.	Planning guide has a few parts completed..	Planning guide is empty.
Written Response 1:	Response accurately describes the purpose, functionality, and inputs/outputs of the app.	Response mostly describes the purpose, functionality, and inputs/outputs of the app.	Response is not complete, but does describe the purpose, functionality, or inputs/outputs of the app.	Response does not address the prompt in any way or is blank.
Written Response 2:	Response fully describes an idea or recommendation provided by a partner / peer and how it improved the app.	Response mostly describes an idea or recommendation provided by a partner / peer and how it improved the app.	Response is not complete, but does describe some of the work with a partner.	Response does not address the prompt in any way or is blank.

Check the  
rubric!

## Step 8: Finish your app!

Submit

# Wrap Up





Name(s) \_\_\_\_\_ Period \_\_\_\_\_ Date \_\_\_\_\_ **C O D E**

### App Development Planning Guide

**Project Description**  
For this project you will work with a partner. Together you will create an app that teaches your classmates about any topic you both find interesting. Along the way you'll learn how to use many of the features of App Lab as well as skills that will help you when building more apps throughout this class.

**You will submit**

- Your final app
- This completed project-planning guide

**App Requirements**

- Uses at least three screens
- Includes examples of images, audio, and text
- A clear and easy to navigate user interface
- Cleanly communicates information about your topic
- Code is cleanly written and free of errors

**Steps**

- Collaborate with your partner to pick a topic you are both interested in
- Interview classmates to identify what they already know about the topic
- Design your app's user interface using this planning guide
- Design and program your app in App Lab
- Collect feedback from your classmates and update your app
- Share your final app with the class

**Investigate**

Step 1. Brainstorm Topic Ideas: Your app can teach your classmates about any topic you and your partner agree on. Your topics could be a hobby, something you've always been interested in, a piece of your personal history, or just something you think your classmates should learn more about.

Write down three ideas for a topic that you brainstorm individually:

Idea 1: \_\_\_\_\_

Idea 2: \_\_\_\_\_

Idea 3: \_\_\_\_\_

## Reflection:

Complete the reflection section of the Planning Guide before turning it in.

# **Unit 3 - Lesson 11**

# **Assessment Day**

# Activity



# Unit Assessment

▼ Unit Assessment



# Gallery Walk

One student per group displays their app on the computer.

Groups rotate around the room running the different apps.

You can leave comments on sticky notes at each station.