Unit 6 Lesson 1

Innovations in Computing

Resources

Name(s)____ Period ______ Date __________________

Activity Guide - Computing Innovations

Innovation Research

Choose one of the following topics, and research the latest innovations in computing hardware. The goal here is to find the **most recent** innovative computing devices within your chosen topic. Keep an eye out in particular for devices that don't *look* like what you might expect a computer to be.

Topics (check the one you've selected)

- ❏ **Wearable Technology** (eg. clothing, jewelry, or accessories with built-in computers)
- ❏ **Health and Safety** (eg. devices that treat disease, track your health, or protect users from danger)
- ❏ **Agriculture** (eg. technology to improve the effectiveness, sustainability, or efficiency of farming)
- ❏ **Manufacturing** (eg. advancements in rapid prototyping, industrial robotics, and the production of goods)
- ❏ **Art and Design** (eg. interactive art or public installations)
- ❏ **Smart Home** (eg. devices that allow you to interact with your thermostat, locks, or lights using computers)

Researching your Topic

With your chosen topic as guidance, go online to research recent innovative computing devices within that topic. Try to find a product that you think is both innovative (in that it's attempting to solve a new problem, or an old problem in a new way) and personally interesting. Visit Code Studio for some recommended sites to kick off your research, as well as more detailed descriptions of each of the topics. As you do your research, consider checking out some of the crowdfunding sites (such as Kickstarter or Indiegogo) to find products that haven't even been released yet!

Use the space below to record notes about interesting products you find, patterns that you're seeing, or problems within your chosen topic that people are trying to address.

Research Notes

An Innovative Solution

Based on the research your group did on the last page, select **one** of the devices you found to focus on. Answer the following questions for your chosen device.

You may need to head back online to gather more details about your chosen device.

What Problem Does it Solve?

This is probably the main sales pitch of the product - why do the creators think this is useful?

What Is Innovative About It?

What makes this device different or better than other solutions out there?

How Do You Interact With It?

Focusing on the Input and Output elements of our model for a computer, how does this device take input from the user, and how does it display output? Try to be as specific as possible.

How Could You Improve It?

What are some changes that could make this device better? Are there common complaints, or clear issues that you might be able to address?

Unit 6 Lesson 2

Designing Screens with Code

Resources

Unit 6 Lesson 3

The Circuit Playground

Resources

Unit 6 Lesson 4

Input Unplugged

Resources

Activity Guide - Input and Events: Program

Your Role: Program

Version A: Asking for Input

Your Role
Your role is the **Program**. It's your job to ask each of the Inputs for their current value whenever the code says to. If the current value of a given Input matches your conditional, then **you perform** whatever behavior is in the conditional.

Other Roles
Each of the other members of your group is an Input. The Inputs will each draw a card every time the draw loop runs, which they will then show to you if asked.

Rules

- Read through the Program aloud
- When you reach an input block, ask the appropriate input for their current value
- The Input will show you their card. If it matches your condition, perform the associated action

Run the program to the right a few times, changing roles each time so that every group member gets a chance to be the Program. Once everyone is comfortable with the system, try coming up with some programs of your own.

Program	Action
`function draw() {`	Start loop, each Input takes one card from the deck

`if (inputOne() == "red") {`
`clap()`
`}`
	Program asks Input One for input
	Input One shows their card to the Program.
	If the card is red, Program claps.

`if (inputTwo() == "red") {`
`stand()`
`}`
	Program asks Input Two for input
	Input Two shows their card to the Program.
	If the card is red, Program stands.

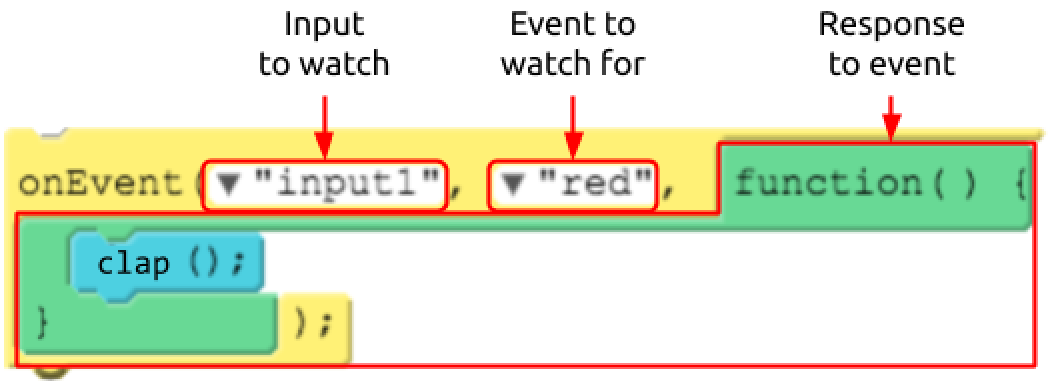
`if (inputThree() =="red") {`
`sit()`
`}`
	Program asks Input Three for input
	Input Three shows their card to the Program.
	If the card is red, Program sits.

`}`	Return to start

Your Role: Program

Version B: Input Events

In this version the Program *does not* continually ask for values and check them with conditionals. Instead, the Program assigns each Input an *Event* to watch for.



Once an Input knows what Event to watch for, the program no longer has to worry about it. It's now the Input's job to report back to the Program if they see the event they've been assigned. A red card being drawn is considered a "red" event, while a black card could be a "black" event.

Rules

- Inputs draw cards off the top of the deck continually at a rate of roughly one card per second
- The Program reads the code aloud
- When the Program reads an `onEvent()` block, tell the appropriate Input what Event to watch for, and how to respond
- When an Input draws a card that matches an event that they're watching, they tell the Program how to respond.

Program	Action
	All Inputs begin drawing cards from the deck once a second

`onEvent("inputOne", "red", function() {`
`    clap()`
`})`
	Input One starts looking for red cards. Whenever Input One draws a red card is drawn they tell Program to clap.

`onEvent("inputTwo", "black", function() {`
`    jump()`
`})`
	Input Two starts looking for black cards. Whenever Input Two draws a black card they Program to jump.

`onEvent("inputThree", "red", function() {`
`    snap()`
`})`
	Input Three starts looking for red cards. Whenever Input Three draws a red card, they tell Program to snap.

As before, run the program a few times, changing up the roles so everyone gets a chance to be the Program. Once you've got the hang of it you can try speeding up the pace of card drawing or writing programs of your own to test. If you write a new program, make sure that the Program is the one who still "runs" all of the code and performs the action.

Your Role: Input 1

Version A: Asking for Input

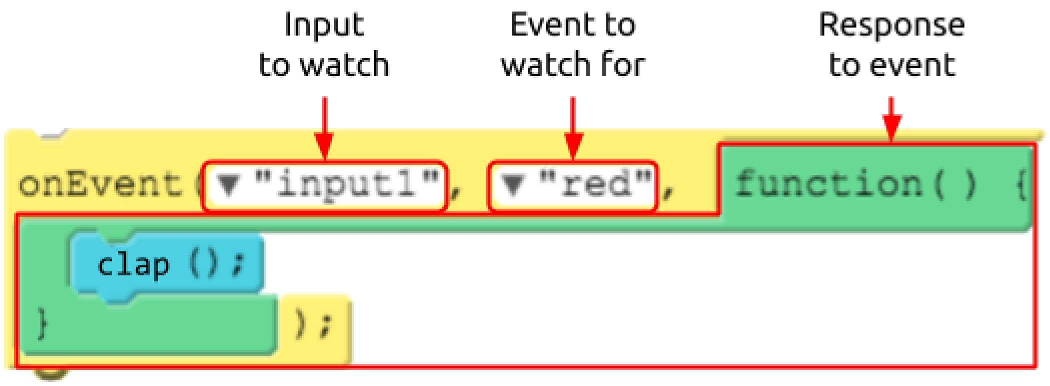
Your Role
Your role is the **Input 1**. It's your job to represent an input to the program by drawing a new card from the deck every time the draw loop runs, showing it to the Program if asked. The program will then decide whether to perform an action based on your input.

Other Roles
- **Program** - Runs the main program and asks for input when appropriate.
- **Inputs 2 and 3** - At the beginning of each draw loop, selects a card from the deck.

Rules
- Every time the draw loop is run, each Input draws a new card from the deck.
- If the Program asks you for input, show your card.

Your Role: Input 1

Model B: Input Events



```
onEvent("input1", "red", function() {
  clap();
});
```

In this version the Program *does not* continually ask for values and check them with conditionals. Instead, the Program assigns each Input an *Event* to watch for.

Once an Input knows what Event to watch for, the program no longer has to worry about it. It's now the Input's job to report back to the Program if they see the event they've been assigned. A red card being drawn is considered a "red" event, while a black card could be a "black" event.

Rules

- When the program begins, start drawing cards from the deck at a rate of roughly one per second
- If the Program assigns you an Event, write down the details in your Events to Watch table
- If you draw a card that matches one of the Events in your table, tell the Program to do whatever is in the *Response* column

Input 1 Events to Watch

Event	Response

As before, run the program a few times, changing up the roles so everyone gets a chance to be the Program. Once you've got the hang of it you can try speeding up the pace of card drawing or writing programs of your own to test. If you write a new program, make sure that the Program is the one who still "runs" all of the code and performs the action.

Your Role: Input 2

Version A: Asking for Input

Your Role
Your role is the **Input 2**. It's your job to represent an input to the program by drawing a new card from the deck every time the draw loop runs, showing it to the Program if asked. The program will then decide whether to perform an action based on your input.

Other Roles

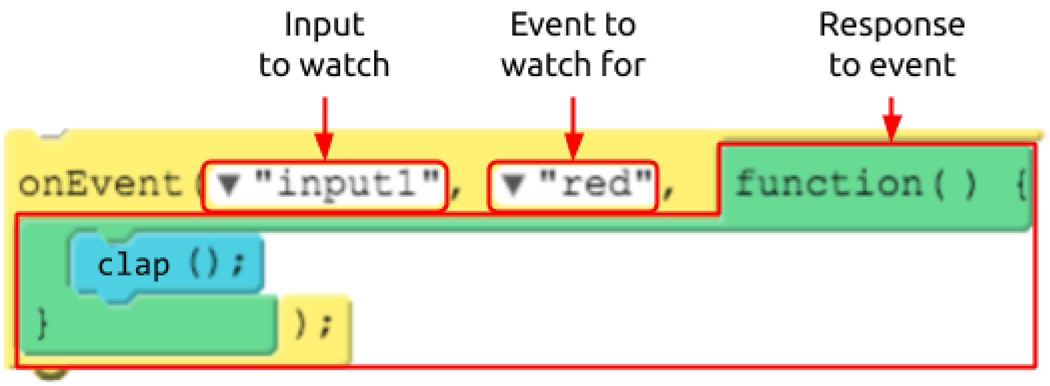
- **Program** - Runs the main program and asks for input when appropriate.
- **Inputs 1 and 3** - At the beginning of each draw loop, selects a card from the deck.

Rules

- Every time the draw loop is run, each Input draws a new card from the deck.
- If the Program asks you for input, show your card.

Your Role: Input 2

Model B: Input Events



```
onEvent("input1", "red", function( ) {
  clap();
});
```

In this version the Program *does not* continually ask for values and check them with conditionals. Instead, the Program assigns each Input an *Event* to watch for.

Once an Input knows what Event to watch for, the program no longer has to worry about it. It's now the Input's job to report back to the Program if they see the event they've been assigned. A red card being drawn is considered a "red" event, while a black card could be a "black" event.

Rules

- When the program begins, start drawing cards from the deck at a rate of roughly one per second.
- If the Program assigns you an Event, write down the details in your Events to Watch table.
- If you draw a card that matches one of the Events in your table, tell the Program to do whatever is in the *Response* column.

Input 2 Events to Watch

Event	Response

As before, run the program a few times, changing up the roles so everyone gets a chance to be the Program. Once you've got the hang of it you can try speeding up the pace of card drawing or writing programs of your own to test. If you write a new program, make sure that the Program is the one who still "runs" all of the code and performs the action.

Your Role: Input 3

Version A: Asking for Input

Your Role
Your role is the **Input 3**. It's your job to represent an input to the program by drawing a new card from the deck every time the draw loop runs, showing it to the Program if asked. The program will then decide whether to perform an action based on your input.

Other Roles

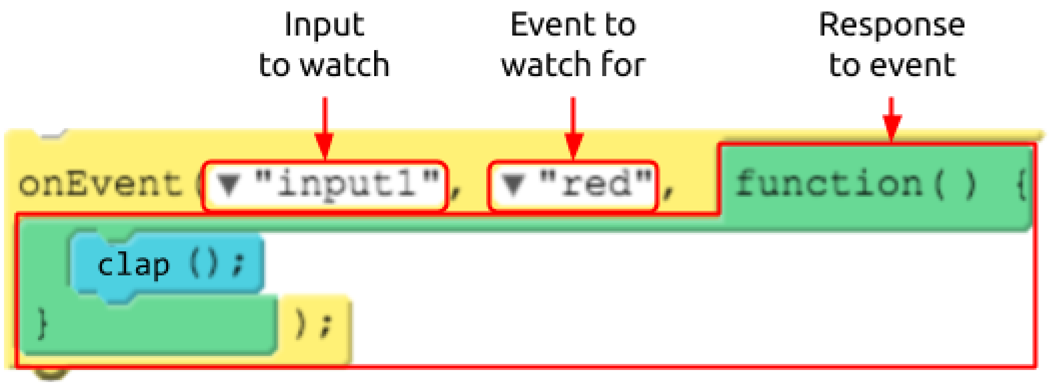
- **Program** - Runs the main program and asks for input when appropriate.
- **Inputs 1 and 2** - At the beginning of each draw loop, selects a card from the deck.

Rules

- Every time the draw loop is run, each Input draws a new card from the deck.
- If the Program asks you for input, show your card.

Your Role: Input 3

Model B: Input Events



```
onEvent("input1", "red", function() {
  clap();
});
```

In this version the Program *does not* continually ask for values and check them with conditionals. Instead, the Program assigns each Input an *Event* to watch for.

Once an Input knows what Event to watch for, the program no longer has to worry about it. It's now the Input's job to report back to the Program if they see the event they've been assigned. A red card being drawn is considered a "red" event, while a black card could be a "black" event.

Rules

- When the program begins, start drawing cards from the deck at a rate of roughly one per second.
- If the Program assigns you an Event, write down the details in your Events to Watch table.
- If you draw a card that matches one of the Events in your table, tell the Program to do whatever is in the *Response* column.

Input 3 Events to Watch

Event	Response

As before, run the program a few times, changing up the roles so everyone gets a chance to be the Program. Once you've got the hang of it you can try speeding up the pace of card drawing or writing programs of your own to test. If you write a new program, make sure that the Program is the one who still "runs" all of the code and performs the action.

Unit 6 Lesson 5

Board Events

Resources

Unit 6 Lesson 6

Getting Properties

Resources

Unit 6 Lesson 7

Analog Input

Resources

Unit 6 Lesson 8

The Program Design Process

Resources

Name(s)____ Period ______ Date ____________________

Project Guide - Emoji Race

Overview

Building a larger piece of software like a game can quickly get complex. Starting with a plan can help you stay organized and identify issues ahead of time. A lot of the work you do here will make it much easier to keep track of what you need to do once you begin writing your actual code.

Program Goal and Design

Start by thinking about what your game actually does. How will the user interact with it? How does it communicate information to the player? What will make it fun, interesting, or relevant to the player?

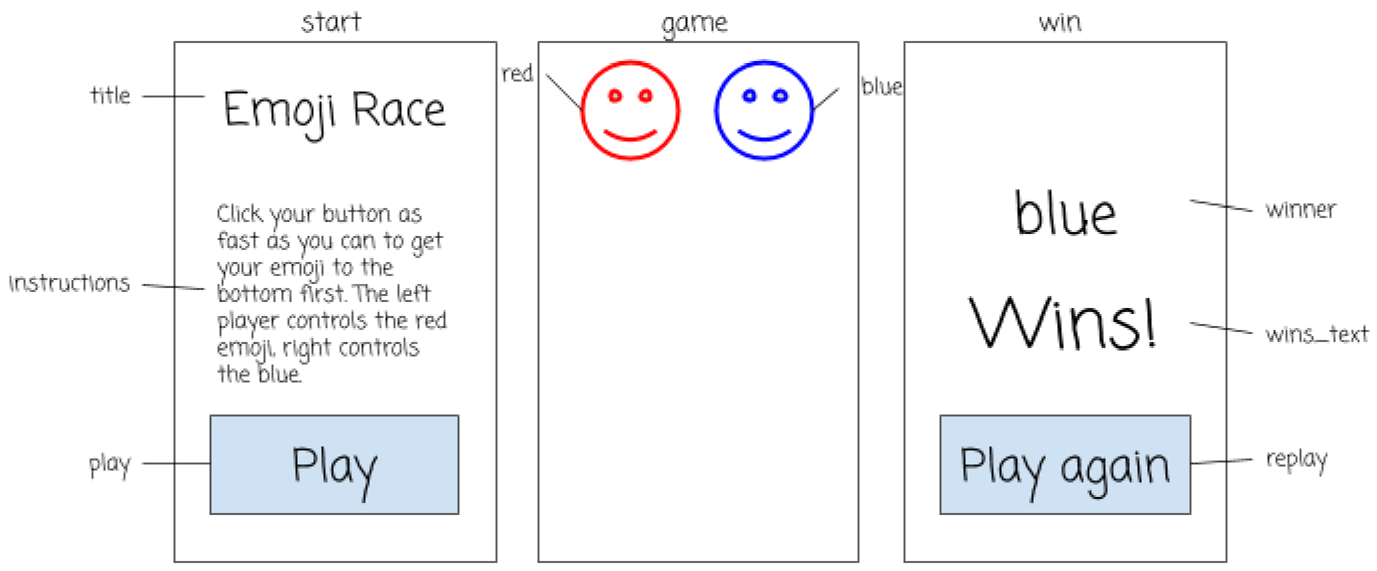
Describe Your Program

In a couple of sentences describe the program you are going to build and how it will work.

Two players are competing clicker race. Each player needs to click one of the Circuit Playground buttons as fast as possible to move their emoji to the bottom of the screen. Whoever gets to the bottom first. The buzzer plays a high note if the red player wins and a low note if the blue player wins..

Draw Your Screen(s)

Draw a quick sketch of the screen(s) you'll need. What design elements will you use? What should their IDs be?


start
game
win
title
Emoji Race
instructions
Click your button as fast as you can to get your emoji to the bottom first. The left player controls the red emoji, right controls the blue.
play
Play
red
blue
blue
winner
Wins!
wins_text
Play again
replay

Circuit Playground

Which components of the Circuit Playground does this program use? Make sure that you are using at least one input (eg buttons and sensors) and one output (eg LED or buzzer).

Board Component	What it is Used For
buttonL	Player one clicks buttonL to move the red player down
buttonR	Player two clicks buttonR to move the blue player down
buzzer	Plays when the game is over

Events and Functions

Using the description of your program above, figure out what events you'll need to respond to and which functions

Events

In the table below list information about all of the events that your program will use, and what will happen when they are triggered

Name or ID	**Event Type** (eg "click")	**Description** (What happens when this event occurs?)
buttonL	*press*	*Move the red emoji down the screen by 10 pixels and check to see if they have reached the bottom*
buttonR	*press*	*Move the blue emoji down the screen by 10 pixels and check to see if they have reached the bottom*
"play"	*click*	*Reset players to the top of the screen and change to the "game" screen*
"replay"	*click*	*Reset players to the top of the screen and change to the "game" screen*

Functions

Your events shouldn't have a lot of complex code. Instead, break your program up into the major steps you'll need for it to work. The different behaviors you described in your events should help you decide what these steps should be.

Function name	**Parameters** (Inputs to the function)	**How It Changes During the Program** (What's the starting value, when will it change?)
startGame()		Resets the players to the top of the screen and changes to the "game" screen
movePlayer()	player	Moves the specified player 10 pixels down the screen
checkWin()	player	Checks to see if the specified player has reached the bottom.. If so, set the text of "winner" to the player, switch to the "win" screen, and buzz

Additional Notes

Use this area to take any extra notes that you might need to complete the program. This could include any variables that you might need, hardware setup for the board, or resources that you'll need to find (like images, sounds, etc), or ideas for more features that you want to explore.

In the movePlayer() and checkWin() functions, I'm going to use a variable player_y to store the player's current y position so that I can add 10 to it and reset.

I still need to figure out how to keep the players from moving their emojis when they shouldn't (like when the "start" or "win" screens are showing).

It would be cool if the emojis started out frowning and then turned into meh and eventually smiling as they move down the screen.

Unit 6 Lesson 9

Project: Make a Game

Resources

Name(s)____ Period ______ Date ___________________

Project Guide - Make a Game

Overview

Building a larger piece of software like a game can quickly get complex. Starting with a plan can help you stay organized and identify issues ahead of time. A lot of the work you do here will make it much easier to keep track of what you need to do once you begin writing your actual code.

Program Goal and Design

Start by thinking about what your game actually does. How will the user interact with it? How does it communicate information to the player? What will make it fun, interesting, or relevant to the player?

Describe Your Program

In a couple of sentences describe the program you are going to build and how it will work.

Draw Your Screen(s)

Draw a quick sketch of the screen(s) you'll need. What design elements will you use? What should their IDs be?

Circuit Playground

Which components of the Circuit Playground does this program use? Make sure that you are using at least one input (eg buttons and sensors) and one output (eg LED or buzzer).

Board Component	What it is Used For

Events and Functions

Using the description of your program above, figure out what events you'll need to respond to and which functions

Events

In the table below list information about all of the events that your program will use, and what will happen when they are triggered.

Name or ID	Event Type (eg "click")	Description (What happens when this event occurs?)

Functions

Your events shouldn't have a lot of complex code. Instead, break your program up into the major steps you'll need for it to work. The different behaviors you described in your events should help you decide what these steps should be.

Function name	**Parameters** (Inputs to the function)	**What does it do?**

Additional Notes

Use this area to take any extra notes that you might need to complete the program. This could include any variables that you might need, hardware setup for the board, or resources that you'll need to find (like images, sounds, etc), or ideas for more features that you want to explore.

Name(s)____ Period ______ Date ________________

Rubric - Board Output Project

Board Output Rubric

Evaluate your program according the following criteria. Explain where in your code each criteria can be found in the comments column.

Criteria	Yes/No	Comments
The program uses at least two event handlers to respond to user input		
The program is logically organized using functions		
The program uses board elements for output		
The program code makes use of whitespace, indentation, and comments to aid the reader		
The program is usable and works as intended		
The project guide has been fully and neatly completed		

Practices Reflection

Practice	Things to Celebrate	Things to Work On
Problem Solving		
Persistence		
Creativity		
Collaboration		
Communication		

Unit 6 Lesson 10

Arrays and Color LEDs

Resources

Unit 6 Lesson 11

Making Music

Resources

Unit 6 Lesson 12

Arrays and For Loops

Resources

Unit 6 Lesson 13

Accelerometer

Resources

Unit 6 Lesson 14

Functions with Parameters

Resources

Unit 6 Lesson 15

Circuits and Physical Prototypes

Resources

Name(s)_______________________________________ Period ______ Date ___________________

Project Guide - Smart Bike Prototype

Overview

Designing a computing device that combines hardware and software requires a good deal of preparation. Starting with a clear plan can help you stay organized and identify issues ahead of time. A lot of the work you do here will make it much easier to keep track of what you need to do once you begin creating your device, both the physical and software components.

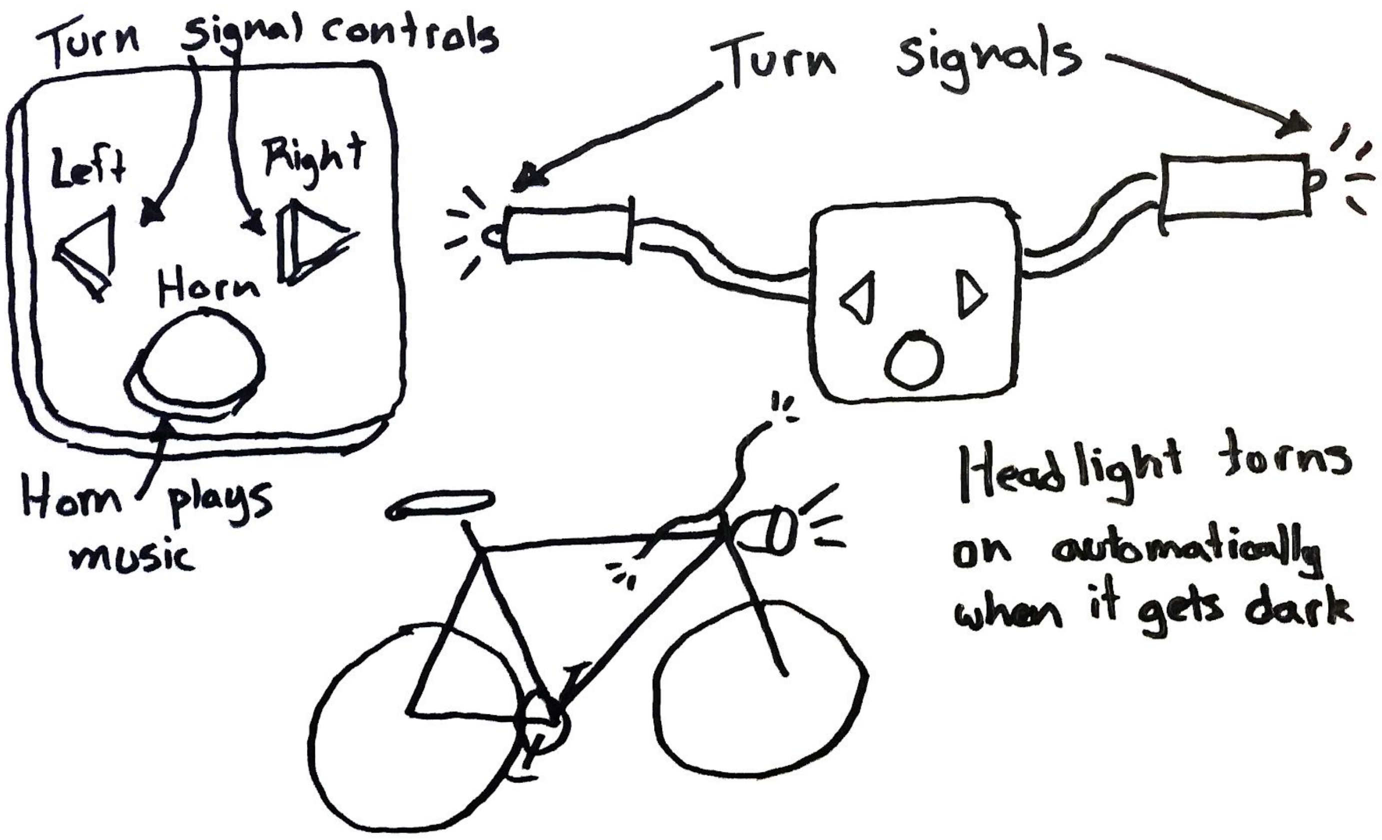
Device Goal and Design

Start by thinking about what problem your device is going to solve. How will the user interact with it? How does it communicate information back to the user? What shape will it take?

Sketch and Describe Your Device

Describe your device and roughly sketch out the main elements. Don't worry about making it pretty.

The smart bike controller is a computer that can be attached to the handlebars of any bike to turn a regular bike into a "smart" bike. It has controls to activate turn signals on the ends of the handlebars, a horn that can play a customizable tone or song, and automatic headlights that turn on when it gets dark. We might also include a speedometer and a safety sensor that detects when a car is too close.



Inputs

What inputs will your prototype need to function? What will they be used for?

Input type	What it is used For
Buttons (at least 3)	Activating turn signals and horn
Light sensor	Turning on headlight automatically

Outputs

What outputs will your prototype need? What will they communicate to the user?

Output Type	What does it communicate?
Handlebar LEDs	Blinking when the rider intends to turn. May also be used as an alert when cars get too close.
Buzzer	Acts as a horn. Could also be used as an alert when cars get too close.
Headlight	Bright LEDs to improve vision and visibility in the dark

Processing

How will you use the inputs to decide what the outputs should be? Break the program up into the major steps you'll need for it to work. The different behaviors you described in your events should help you decide what these steps should be.

Function name	**Parameters** (Inputs to the function)	**How It Changes During the Program** (What's the starting value, when will it change?)

Additional Notes

Use this area to take any extra notes that you might need to complete the program. This could include any variables that you might need, hardware setup for the board, or resources that you'll need to find (like images, sounds, etc), or ideas for more features that you want to explore.

Unit 6 Lesson 16

Project: Prototype an Innovation

Resources

Name(s)____ Period ______ Date ___________________

Project Guide - Innovation Prototype

Overview

Designing a computing device that combines hardware and software requires a good deal of preparation. Starting with a clear plan can help you stay organized and identify issues ahead of time. A lot of the work you do here will make it much easier to keep track of what you need to do once you begin creating your device, both the physical and software components.

Device Goal and Design

Start by thinking about what problem your device is going to solve. How will the user interact with it? How does it communicate information back to the user? What shape will it take?

Sketch and Describe Your Device

Describe your device and roughly sketch out the main elements. Don't worry about making it pretty.

Inputs

What inputs will your prototype need to function? What will they be used for?

Input type	What it is used For

Outputs

What outputs will your prototype need? What will they communicate to the user?

Output Type	What does it communicate?

Processing

How will your program turn input into output? You won't be able to do everything at once. Instead, break your program up into the major steps you'll need for it to work. The different behaviors you described in your events should help you decide what these steps should be.

Function name	**Parameters** (Inputs to the function)	**What does it do?**

Develop Your Prototype

Once your teacher has approved your design, use the materials you are given to develop your prototype.

Test Your Prototype

Test your prototype with multiple users and, if time allows, incorporate feedback from your testing.

Reflect

What part of your project are you most proud of? ______________________________

Why? ______________________________

____

If you had more time, what improvement would you make to your innovation?

____

____

Peer Review - Prototype an Innovation

Pre-Review

Creator's Name: ________________________________

One thing I want feedback on is…____

____

Reviewer Section

Reviewer's Name: ________________________________

Questions	Rating	Comments
I can understand how to use the innovation. I can understand what the innovation does, how to use it, and what problem it's attempting to solve.	✓ ✗	
The code of the program is clean and easy of read. I can easily read and understand how the code for this program works.	✓ ✗	
The code of the program uses design elements, events, and loops appropriately. The code of this program makes use of the programming constructs we learned.	✓ ✗	

Free Response Feedback

I like… ____

____

I wish… ____

____

What if… ____

____

Creator’s Reflection

1. What piece of feedback was most helpful to you? Why?

2. What piece of feedback surprised you the most? Why?

3. Based on feedback, what changes will you make to your innovation?

Name(s)_______________________________________ Period ______ Date ________________

Rubric - Prototype an Innovation Project

Project-Specific Rubric

Criteria	Yes/No	Comments
The innovation uses at least one of the board inputs (buttons, sensors, etc)		
The innovates uses at least one of the board outputs (buzzer, leds, etc)		
The innovation includes at least one UI element with an appropriate event handler		
The program works as intended		
The program code makes use of whitespace, indentation, and comments to aid the reader		
The project guide has been fully and neatly completed		
The peer review provides useful and constructive feedback		
Peer review feedback has clearly been incorporated into the final version of the project.		

Practices Reflection

Practice	Things to Celebrate	Things to Work On
Problem Solving		
Persistence		
Creativity		
Collaboration		
Communication		