Name(s)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Period \_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
|  | **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
* Clearly 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 and Reflect Phase

**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: How to do woodturning. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Idea 2: How to code some basic C. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Idea 3: A quiz on computer science topics from Unit 3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Step 2. Choose One Topic:** Now talk through your ideas with your partner. Together pick a topic both of your 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: How to cde some basic C.

The team (I) will talk about how to set up a basic main.c file. I will explain what header files are and what functions are. This wont go into much detail since I only have a few pages and little time to complete this, but at the end a working basic program will be made. People will be able to follow along and learn the very basic of coding in C.

**Step 3. Interview 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

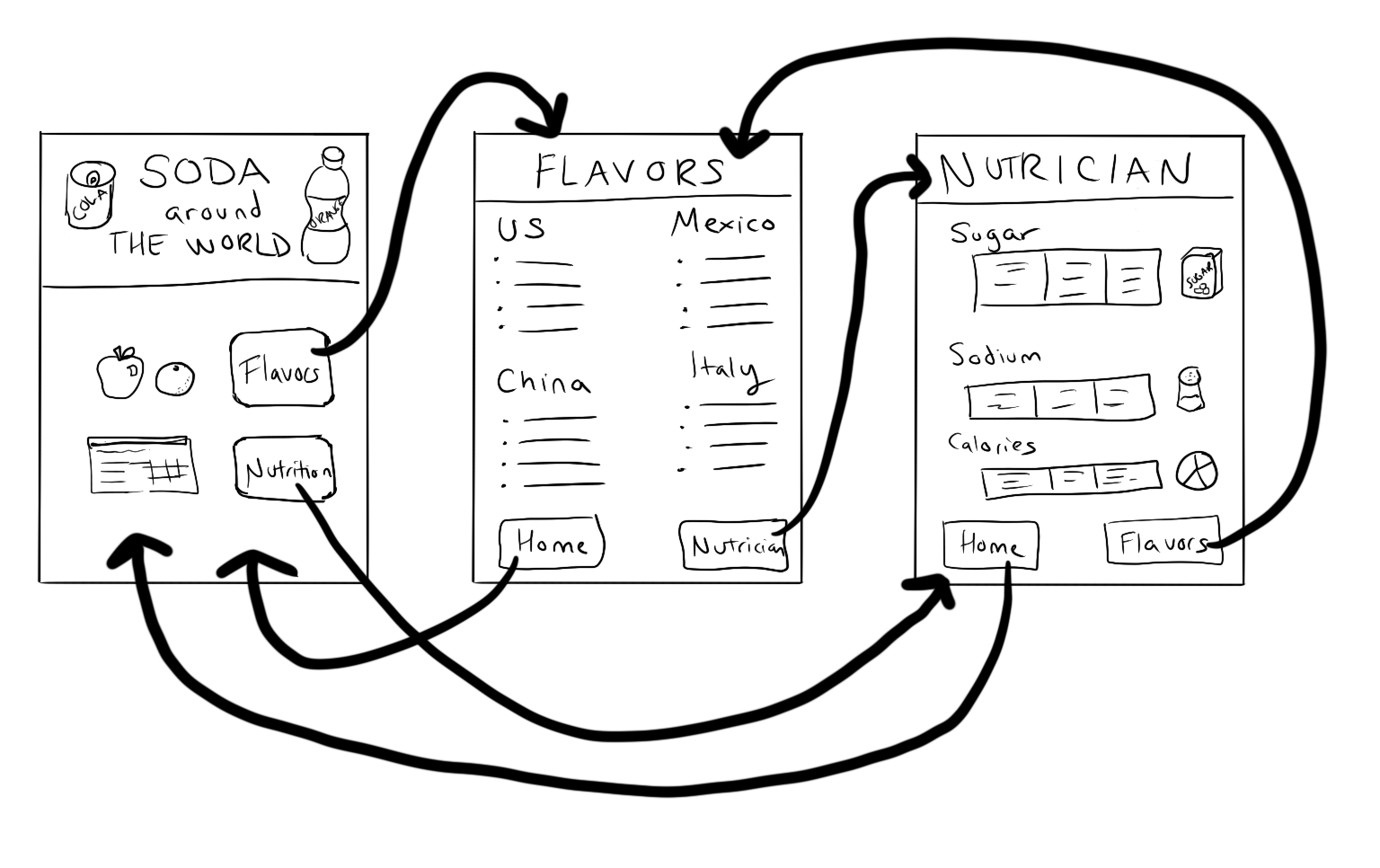
\* I used my parents

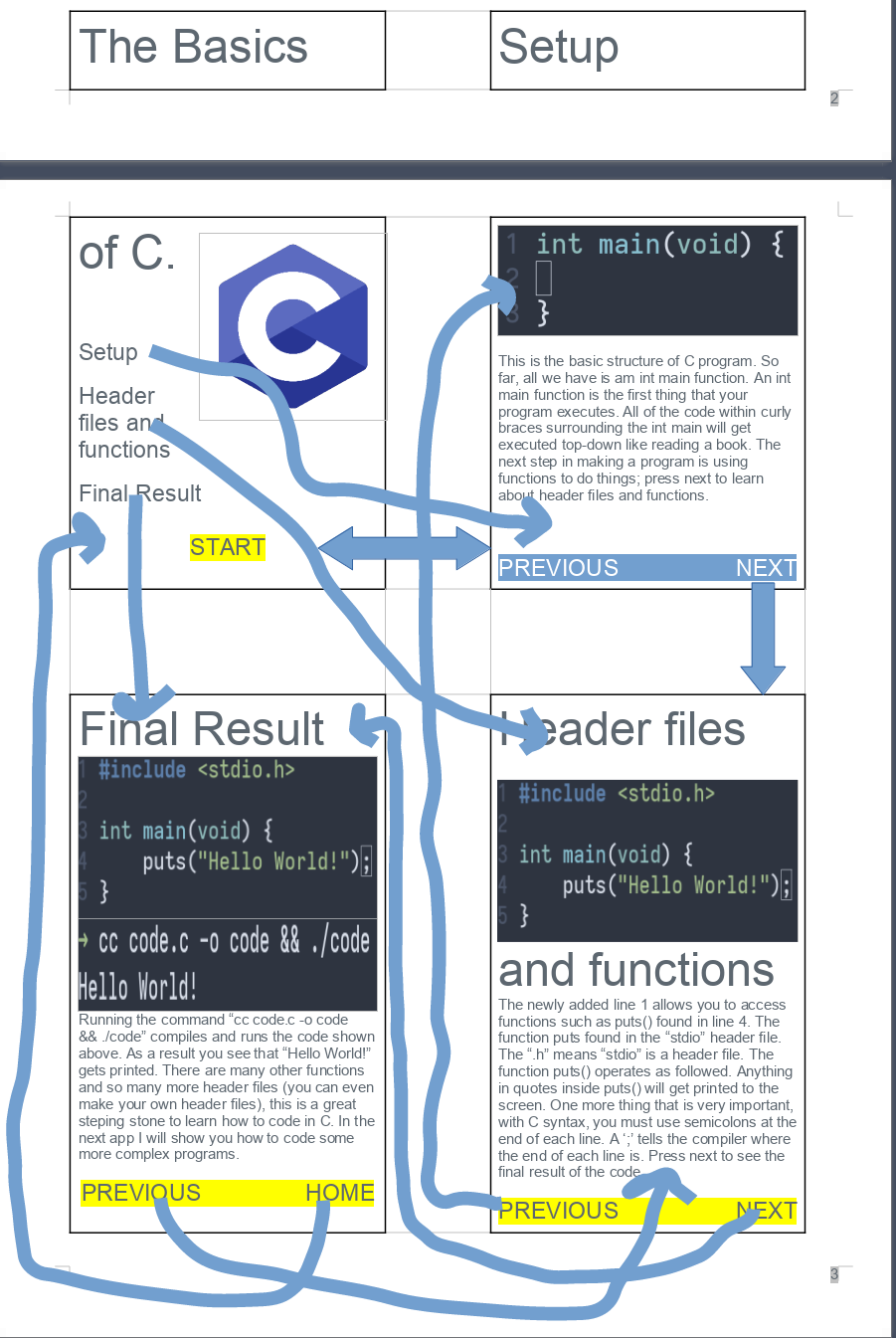
|  |  |  |
| --- | --- | --- |
| **Name** | **What do they already know about your topic?** | **What do they need or want to learn about your topic?** |
| Alan Sikarskie | My father already knows that C is a low level imperative programming langurage made by Dennis Ritchie. My father has had some experience coding in C. | My father would like to relearn the basic syntax structure and how to use functions inside the main function. |
| Amy Sikarskie | My mother does not know anything about the progtramming langurage C, except for the fact that I have been coding in it for a liittle bit. | My mother would like to lean for the first time how to print hello world and what header files are/ what header files are used for. |

My targeted audience is people who want to learn how to code in C. What I intended for them to do with the app is to learn the basics of C coding. I intend for the audience to learn how to make a “Hello World!” program coded in C. The audience would want to use the app because learning how to code will help you in life even if it's not your job. Knowing how to code will allow you to create basic scripts to do tasks that would otherwise take you hours to do.

## Design Phase

**Step 4. Create a Program Specification:** Based on your research you identified **requirements** for what your app must teach your classmates. On the next page you should draw a **specification** that shows how your app will actually run to meet those requirements. 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.





## Building Phase

**Step 5. Start Building Your App:** Work with your partner to build your app. Along the way make sure you:

* Use the program specification you drew as a starting point, but it’s OK to update as you go.
* Use pair programming
* Use your debugging skills to check that your app is working

Before you begin to code, fill out the chart below for any Event Handlers in your program:

|  |  |  |
| --- | --- | --- |
| Element ID | Action | What happens? |
| *“dogButton”* | *“click”* | *A picture of a dog appears*  *The background of the screen changes to green* |
| “mainScreenC” | “none” | This is the main screen of the C section in the app. Here are buttons to navigate the app. |
| “setupScreenC” | “none” | This is the setup screen of the C section in the app. Here are buttons to navigate the app. |
| “headerScreenC” | “none” | This is the header screen of the C section in the app. Here is information about header files and functions. |
| “finalScreenC” | “none” | This is the final screen of the C section in the app. Here is the final result of the C program with information about the future. |
| “cLogoImageC” | “none” | Nothing changes if you press it, but it shows the C coding logo/symbol. |
| “mainTitleC” | “none” | Nothing changes if you press it, but it shows the main title of the C section in the app. |
| “setupButtonC” | “click” | Once you click the button, it will send you to the app's C setup screen/setup page. |
| “headerButtonC” | “click” | Once you click the button, it will send you to the header files and functions screen/setup page of the C section in the app. |
| “finalResultButtonC” | “click” | Once you click the button, it will send you to the app's final result screen/setup C page. |
| “startButtonC” | “click” | Once you click the button, it will send you to the app's setup screen/setup C page. |
| “templateCodeImageC” | “none” | Nothing changes if you press it, but it shows a C code program's basic set up. |
| “setupInfoC” | “none” | Nothing changes if you press it, but it shows information about C code's basic setup structure. |
| “setupBackButtonC” | “click” | Once you click the button, it will send you to the app's C home/main screen/setup page. |
| “setupNextButtonC” | “click” | Once you click the button, it will send you to the header files and functions C screen/setup page of the app. |
| “codeImageHeaderC” | “none” | Nothing changes if you press it, but it shows the C code discussed on the page. |
| “headerInfoC” | “none” | Nothing changes if you press it, but it shows information about the basic structure of headers and C code functions. |
| “headerBackButtonC” | “click” | Once you click the button, it will send you to the app's C setup screen/setup page. |
| “headerNextButtonC” | “click” | Once you click the button, it will send you to the app's C final result screen/setup page. |
| “codeImageFinalC” | “none” | Nothing changes if you press it, but it shows the C code discussed on the page. |
| “compileImageC” | “none” | Nothing changes if you press it, but it shows the C code discussed being compiled and ran. |
| “headerInfoC” | “none” | Nothing changes if you press it, but it shows information about the C program's final result with information about the future. |
| “finalBackButtonC” | “click” | Once you click the button, it will send you to the C header files and functions screen/setup page of the app |
| “finalNextButtonC” | “click” | Once you click the button, it will send you to the app's C home/main screen/setup page. |
| “c++ButtonC” | “click” | Once you click the button, it will send you to the C++ main page where you will learn the basics of C++. |
| “choiceScreen” | “none” | This is the main screen in the app. Here are buttons to select what coding language the user wants to learn about. |
| “choiceTitle” | “none” | Nothing changes if you press it, but it shows the main title of the app. |
| “choiceText” | “none” | Nothing changes if you press it, but it shows information about what to do once the app has started. |
| “choiceCLogo” | “click” | Nothing changes if you press it, but it shows the C coding logo/symbol. |
| “choiceC++Logo” | “click” | Nothing changes if you press it, but it shows the C++ coding logo/symbol. |
| “finalScreenC++” | “none” | This is the final screen of the C++ section in the app. Here is the final result of the C++ program with information about the future. |
| “compileImageC++” | “none” | Nothing changes if you press it, but it shows the C++ code discussed being compiled and ran. |
| “finalInfoC++” | “none” | Nothing changes if you press it, but it shows information about the C++ program's final result with information about the future. |
| “finalInfoC” | “none” | Nothing changes if you press it, but it shows information about the C program's final result with information about the future. |
| “finalBackButtonC++” | “click” | Once you click the button, it will send you to the C++ header files and functions screen/setup page of the app |
| “finalNextButtonC++” | “click” | Once you click the button, it will send you to the app's C++ home/main screen/setup page. |
| “headerScreenC++” | “none” | This is the header screen of the C++ section in the app. Here is information about header files and functions. |
| “codeImageHeaderC++” | “none” | Nothing changes if you press it, but it shows the C++ code discussed on the page. |
| “headerInfoC++” | “none” | Nothing changes if you press it, but it shows information about the C++ program's final result with information about the future. |
| “headerBackButtonC++” | “click” | Once you click the button, it will send you to the app's C++ setup screen/setup page. |
| “headerNextButtonC++” | “click” | Once you click the button, it will send you to the app's C++ final result screen/setup page. |
| “mainScreenC++” | “none” | This is the main screen of the C++ section in the app. Here are buttons to navigate the app. |
| “mainTitleC++” | “none” | Nothing changes if you press it, but it shows the main title of the C++ section in the app. |
| “c++LogoImageC++” | “none” | Nothing changes if you press it, but it shows the C++ coding logo/symbol. |
| “setupButtonC++” | “click” | Once you click the button, it will send you to the app's C++ setup screen/setup page. |
| “headerButtonC++” | “click” | Once you click the button, it will send you to the header files and functions screen/setup page of the C++ section in the app. |
| “finalResultButtonC++” | “click” | Once you click the button, it will send you to the app's final result screen/setup C++ page. |
| “startButtonC++” | “click” | Once you click the button, it will send you to the app's setup screen/setup C++ page. |
| “cButtonC++” | “click” | Once you click the button, it will send you to the app's main C page. |
| “c++ButtonC” | “click” | Once you click the button, it will send you to the app's main C++ page. |
| “setupScreenC++” | “none” | This is the setup screen of the C++ section in the app. Here are buttons to navigate the app. |
| “templateCodeImageC++” | “none” | Nothing changes if you press it, but it shows a C++ code program's basic set up. |
| “setupInfoC++” | “none” | Nothing changes if you press it, but it shows information about C++ code's basic setup structure. |
| “setupBackButtonC++” | “click” | Once you click the button, it will send you to the app's C++ home/main screen/setup page. |
| “setupNextButtonC++” | “click” | Once you click the button, it will send you to the header files and functions C++ screen/setup page of the app. |

Use the chart to guide you in adding programming statements to your program.

## 

## 

## Testing Phase

**Step 6. Testing & Feedback:** You will need to test your app to make sure it works as expected. To do that find at least two classmates to use your app.

* Ask students to read through your program specification and requirements
* Ask them to use the app and test out the different behaviors included in your specification
* Write down anything you noticed them finding confusing or broken
* Ask them to share anything they recommend improving

|  |  |  |
| --- | --- | --- |
| **Name** | **Things that could be improved based on watching them use the app** | **Improvements this person recommends** |
| Amy Sikarskie | The buttons could have the same height, so it would allow clicking through the pages more accessible. | The text on the screen could be larger and the buttons could be bigger. |
| Alan Sikarskie | The images need to be smaller; they take up too much space and are causing the text to be too small. | The background of the screen and the buttons could have more fun colors. |
| Ms. Gilbert | Nothing. | The main title on the choiceScreen would look better with a white border. |

**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:

I will make the background and buttons more colorful.

Improvement 2 (Optional):

I will make the text on the screen larger and the images smaller.

**Step 8: Complete Your App:** Finish your app!

## 

## Reflection

**Question 1:** Provide a written response that:

* describes the overall purpose of the program
* describes the functionality of your app
* describes the input and outputs of your app

(Approx 150 words)

The app that I created has an overall purpose of teaching the user how to code a "Hello World!" program written in C and C++. The purpose of doing this is to have more people code in C/C++ and to give users a kickstart in their coding career. The app's functionality is straightforward; there are buttons, images, and text on the screen, which are informative. The app has four screens per coding language, one of which is the starting screen. A user input clicks from their mouse; these clicks then outputs a change in the screen. The difference in the screens is switching one screen to another. The four screens have the necessary information about C/C++ coding, and after following the steps in the app, one would be able to code a "Hello World!" program by themselves.

**Question 2:** This project was created using a development process that required you to incorporate the ideas of your partner and feedback from your classmates. Provide a written response that describes one part of your app that was improved through input from EITHER your partner or feedback you received from classmates. Include:

* Who specifically provided the idea or recommendation
* What their idea or recommendation was
* The specific change you made to your app’s user interface or functionality in response to the recommendation
* How you believe this change improved your app

(Approx 150 words)

I used my mother and father for the testing of this app, Alan and Amy Sikarskie. Alan Sikarskie suggested that I should make the app more colorful. When he said that it made sense, the app at that point was very lackluster. I needed to make the app look less boring. I changed the color of the backgrounds and the color of the buttons on each slide. This change improved my app because now people might be more intrigued by the app's look to stick around and learn a thing or two about coding in C and C++. My mother, Amy Sikarskie, suggested that I make the buttons have the same x and y coordinate, so flipping between pages would be more comfortable. I did what she said, and this also improved my app because it made it look more professional; it also allowed for easier use.

## 

## Rubric

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 | The app includes at least one example each of:   * Text * Image * and audio. | The app includes at least one example of two of the following:   * Text * Image * And audio | The app includes at least one example of one of the following:   * Text * Image * And audio | 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 mostly use meaningful IDs | Some screen elements use meaningful IDs | Screen elements do not use meaningful IDs. |
| App Topic | Topic is clearly communicated and explained. | Topic is somewhat clearly communicated and explained. | Topic is not 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. |

PROMPTS :)

Lesson 1:

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

→ Apps are applications and we interact with them through 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, and 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. Apps are used for anything such as navigation, eduction, and entertainment.

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

How does the user interact with the app?

→ Users interact with them through 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, and 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. Apps are used for anything such as navigation, eduction, and entertainment.

What is the overall purpose of the app?

→ To educate the user about various topics such as water conservation, bird facts, how to play four square and more.

Who is the target audience?

→ People who want to learn about the various topic that the app is teaching them.

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

→ My favorite app is the calculator. The user interface has buttons and the inputs are from the buttons. The outputs are the based off the math formulas/problems that I or any user enters.

Lesson 2:

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.

→ I use the common app, the calculator. The user interface has buttons and the inputs are from the buttons. The outputs are the based off the math formulas/problems that I or any user enters.

What elements collect input?

→Buttons collect input for a calculator app.

What elements display output?

→Text is outputed after a user has pressed a button.

Do you think there are elements that can do both?

→Yes such as a button, it can show information, but it can also recieve information as a boolean once it is pressed.

Why is it important for element IDs to have meaningful names?

→ It is important to have meaningful named element ID's because the element ID's help you determine what element you are editing. If the element ID isn't meaningful you might be editing the wrong element ID.

Lesson 3:

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?

→ Yes I have.

What was the problem?

→ The problem was with the buttons and sliders. Whoever designed the app designed it like a website,

not like an app, so the user interface was tough to use.

What do you think the designers didn't understand about you or your needs?

→ I think they used some old code lying around, and I believe they did not spend the correct amount of

time testing because it is very easy to see the problem.

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

→ Talking with users made me realize what I should change and where I should go for the next steps. I

now have a better understanding of how to make it more user friendly.

Lesson 4:

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

→ It is essential to plan out an app's design because, without it, you could miss something or overthink

some details. Planning the system also makes coding the app easier since you already have done the big

concept brainstorming and thinking.

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

→ Yes, I had to make some icons and images smaller because they cluttered the screen once I made it

online. On paper, I was able to fit more than what I was able to fit the screen.

Lesson 5:

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

→ 1. The user interface is hard to use.

2. The text on the screen is hard to read.

3. The app does not let me control the video, audio, and/or graphics.

4. They are not easy to follow / they make no sense

5. The steps are not in order

6. They are impossible to do

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

→ We did not do this so I am unable to answer.

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?

→You would not need to change anything with the human language; English at least does not need to improve. We are able to explain and give a set of instructions perfectly well as it is therefore there is no need to change the language. In our language we already have words for transitions and conditionals; we have words for everything for programming and we can just use those words.

What is the difference between a programming language and natural (every-day) language?

→ Programming is different from natural spoken language in many ways, one way is that in programming the program cannot infer that you are trying to do; if a program sees code it will only run that code and not make any of its own decisions. Programming is also different from natural spoken language because in programs they understand different syntax than speaking. You would never say for parenthesis int i is equal to 0 while i is less than three add one to i say hi, but instead you would say hi hi hi.

Lesson 6:

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

→ Programming is different from natural spoken language in many ways, one way is that in programming the program cannot infer that you are trying to do; if a program sees code it will only run that code and not make any of its own decisions. Programming is also different from natural spoken language because in programs they understand different syntax than speaking. You would never say for parenthesis int i is equal to 0 while i is less than three add one to i say hi, but instead you would say hi hi hi.

What is the difference between a sequential program and an event-driven program?

→ The difference between sequential programs and event driven-programs is that sequential programs will run the same every time, but with event-driven programs may or may not since the user is in control. With event-driven programs a user is able to interact with the program unlike sequential programs where they are not able to do so.

Lesson 7:

What is one thing you really enjoyed about today's activity?

→ I really enjoyed how fast and easy this assignment was. I had a lot of homework in other classes and it was a refresher to not need to spend a super long time on computer science work.

Is there anything that you found confusing or need extra help with?

→ I do not know how to find the music for problem 8, I used a YouTube video and code.org is not playing it for some reason. I do not know how to access their media.

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.

→

\* Is the speaker plugged into a power source?

\* Is the media (phone or computer) working without the speaker?

\* Is the speaker connected to the media device?

\* Does the speaker work with a different media device?

Share any debugging tips you recorded today with your neighbor. Be ready to share with the class.

→

I learned to follow the debugging process found on slide 6:

-------------------------------

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

-------------------------------

Document

As You Go

What have you learned?

What strategies did you use?

What questions do you have?

-------------------------------

Try

Solutions

Make a small change

-------------------------------

Lesson 8:

What makes a good partner?

→ A good partner is someone who helps you work towards accomplishing a common task.

How does Pair Programming help when working on a project? How does it help with the debugging process in particular?

→ Pair programming allows for another set of eyes to help the coder find what the bugs are and how to improve the code. Pair programming helps with debugging because you have another person who knows what you are trying to do and they are actively helping you search for bugs.

Lesson 9:

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?

→ Good feedback is feedback that goes into detail about what you need to improve on / fix as well as what you did well. Good feedback goes into detail while bad feedback doesnt. An example of good feedback is "The program is working for the most part, but there is a problem with X. To solve X you can try Y also try looking at line Z in the code since that is most likely the cause. An example of bad feedback is "ah it kinda works".

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?

→ Others offer a new way of looking at your app and they often have ideas that end users would like to have fixed or added. The value of getting feedback even if you arent finished with the app is that you will not spend hours on something that will get cut after feedback is recieved.

Lesson 10:

There are no prompts for lesson 10 :0.