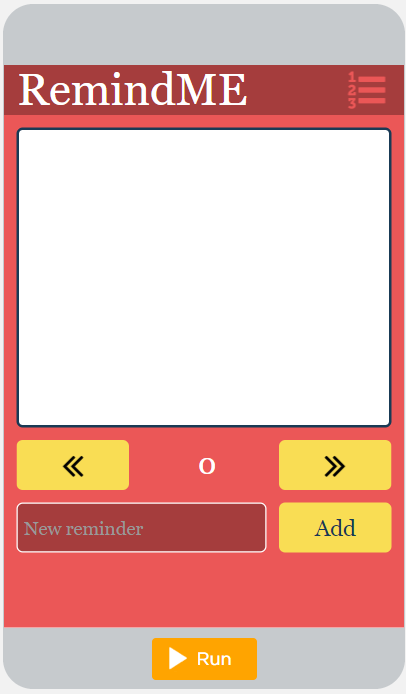
Name(s)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Period \_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
|  | **Activity Guide - Lists Make** |  |



"rightButton"

"reminderOutput"

"addButton"

"reminderInput"

"leftButton"

**Step 1 - Try the app**

* Add several reminders to the app.
* Scroll through your reminders.

Discuss with a Partner

* What programming patterns with lists do you think you'll need to use?

**Step 2 - Plan**

**Lists:** This app uses one list. Give it a name! What does it store?

|  |  |
| --- | --- |
| **List Name** | **What the List Stores** |
|  |  |

**Other Variable:** This app uses one extra variable in addition to the list. What does it store?

|  |  |
| --- | --- |
| **Variable Name** | **What the Variable Stores** |
| *index* |  |

**Conditionals:** Review the List Scrolling Pattern. Consider how you will use this in your app.

|  |  |
| --- | --- |
|  | How does it work? Many App Lab projects run in the following way.   1. The user is interested in seeing items in a list, like a list of favorite foods, or a list of reminders. 2. The app shows the user one item in the list at a time. 3. By clicking buttons (or otherwise interacting with the app) the user can move back and forth through the list, one item at a time, to see every item.   This pattern allows a user to "scroll" through all the items in the list. In order to make it work, create an index variable and a list. Then use the "Counter Pattern with Boundary" pattern to create event handlers to change the value of the index.  This pattern makes sense to use with the updateScreen pattern since you will need at least two buttons for scrolling left and right through the list but afterwards will use the same code to update the screen. |

**Function:** What code may be repeated in your app? What should be included in your function(s)?

* Hint: Check out how the updateScreen() function is set up in the List Scrolling Pattern.

|  |
| --- |
|  |

**Inputs:** What are the inputs for the app? These will all be turned into onEvents.

|  |  |  |
| --- | --- | --- |
| **Input** | **Action** | **Result** |
| *"rightButton"* | *click* | *The reminder stored in the list at the next index is displayed.* |
|  |  |  |
|  |  |  |
|  |  |  |

**Step 3 - Write Your Code**

* Write the code for the app, using your plan above and the comments provided in Code Studio to help
* Step You Can Follow
  + Create all the variables (including the list!) from your tables above.
  + Give your variables a starting value using the assignment operator (=).
  + Create a function that updates the screen.
  + Update the output elements on the screen inside of the function.
  + Create event handlers (onEvent) for the inputs in your table above
  + Inside the event handlers, update the appropriate variable using the List Scrolling Pattern. Then call the update screen function.
  + Use your debugging skills to identify unexpected behavior and fix your program
  + Comment your code as you go, explaining what each event handler and function does
* Extension Ideas
  + Create a button that deletes unwanted items from the list.
  + Change the list scrolling so that it wraps from the end of the list back to the beginning.

**Step 4 - Submit**

Before your submit check the rubric below to make sure your program

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **Extensive Evidence** | **Convincing Evidence** | **Limited Evidence** | **No Evidence** |
| Input | onEvents are created for all the required inputs. | onEvents are created for most of the inputs. | onEvents are created for some of the inputs. | onEvents are not created for any inputs. |
| Storage: Variables and Lists | Variables and lists are created and appropriately used for all pieces of information used in the app. | Variables and lists are created and appropriately used for most pieces of information used in the app. | Some information is stored in variables and lists and appropriately updated throughout the app. | There are no variables or lists which store the necessary information for the app to work correctly. |
| Processing: Lists | The program correctly processes the list for all user interface elements. | The program correctly processes the list for most user interface elements. | The program correctly processes the list for some of the user interface elements. | The program does not include or does not process a list. |
| Code: Functions | A function is used which correctly updates all output elements. The function is called in the appropriate onEvents. | A function is used which correctly updates most of the output elements. The function is called in the appropriate onEvents. | A function is used which updates some of the output elements or the function is only called in some of the appropriate onEvents. | There is no function which updates the screen. |
| Output | The screen correctly displays the current reminders in all instances. | The screen correctly displays the current reminders in most instances. | The screen correctly displays some but not all information. | The screen does not correctly display any stored information. |
| Code runs without errors. | No errors are present in the required code. | One or two errors are present in the required code. | Three or four errors are present in the required code. | More than four errors are present in the required code. |
| Coding Comments | Comments are used to correctly explain the purpose and function of all onEvents and functions. | Comments are used to explain the purpose and function of most onEvents and functions. | Comments are used to explain the purpose and function of some onEvents and functions. | Comments are not present. |