Create Performance Task Template 2018

# 2a.

**Write your 2a response here (<150 words):**

|  |
| --- |
| Hello, welcome to my app showcase. My app is a task app that will help people become more organized in their lives. I will display the features of my app right now. Here is the way someone will add a task to their list. The app will then send them to the editing/info page. This info page will allow them to change different things about the task they have just entered. They will be able to change the task name, change the due date, and change the notes on the task. They will then be able to either delete the task or just close the info page to go back to the main screen. Everything entered by the user will be saved so when the user goes back into the app their tasks will still be present. |

Video: <https://www.youtube.com/watch?v=b8_ZFp5DZA4&feature=youtu.be>

# 2b.

**Write your 2b response here (< 200 words):**

|  |
| --- |
| I started my app off by creating my main UI where the user will be able to add tasks. After getting the UI done I moved to the blocks section and started to add my code to create a popup for the user to enter their tasks and view all the tasks that they have entered. I had difficulty with adding tasks because my list view would not show everything I added but I just forget to switch on/off my variables to the right value. Next, I moved to the Info page that would allow me to change the name of the task, the notes associated with the app and the due date. The hardest part about adding these features was saving the notes to the correct task and keeping them saved in a TinyDB. I ran into some problems with changing pages and sending variables so I developed a procedure that dynamically sets the visibility of vertical arrangements. In each of these vertical arrangements was a different tab of UI. This allowed me to keep all my code on one screen and not have to switch back and forth across different screens. |

# 2c.

**Write your 2c response here (< 200 words)**

|  |
| --- |
| The procedure below is the procedure I created to dynamically switch what vertical arrangement was visible. The first algorithm checks if the variable screen number is equal to 1. If it is then it will set the first vertical arrangement to visible and set every other arrangement set to false. Next, my procedure will check if ScreenNumber is equal to 2. If it is it will make vertical arrangement 2 visible and make any other arrangements invisible. The same algorithm is repeated for vertical arrangement 3. All together this procedure allows me to dynamically change the visible arrangement without massive amounts of code needed every time I want to stitch visible UI. This also allows me to keep all my UI and code on one screen and not have to worry about transferring variables across screens. |

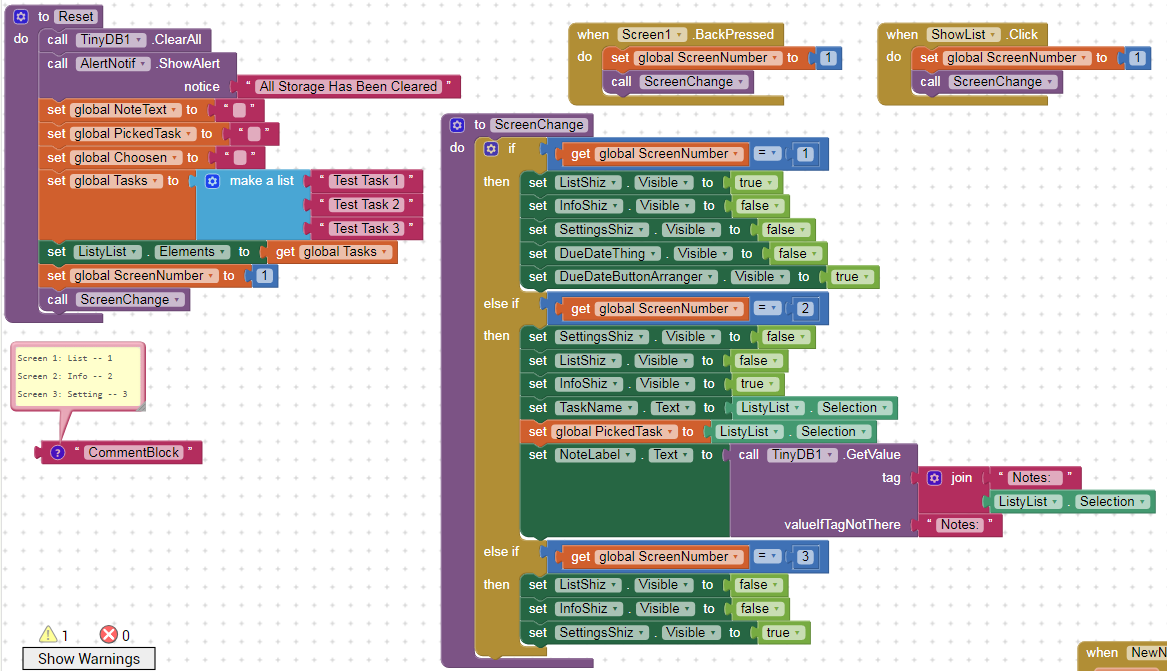
# 2d.

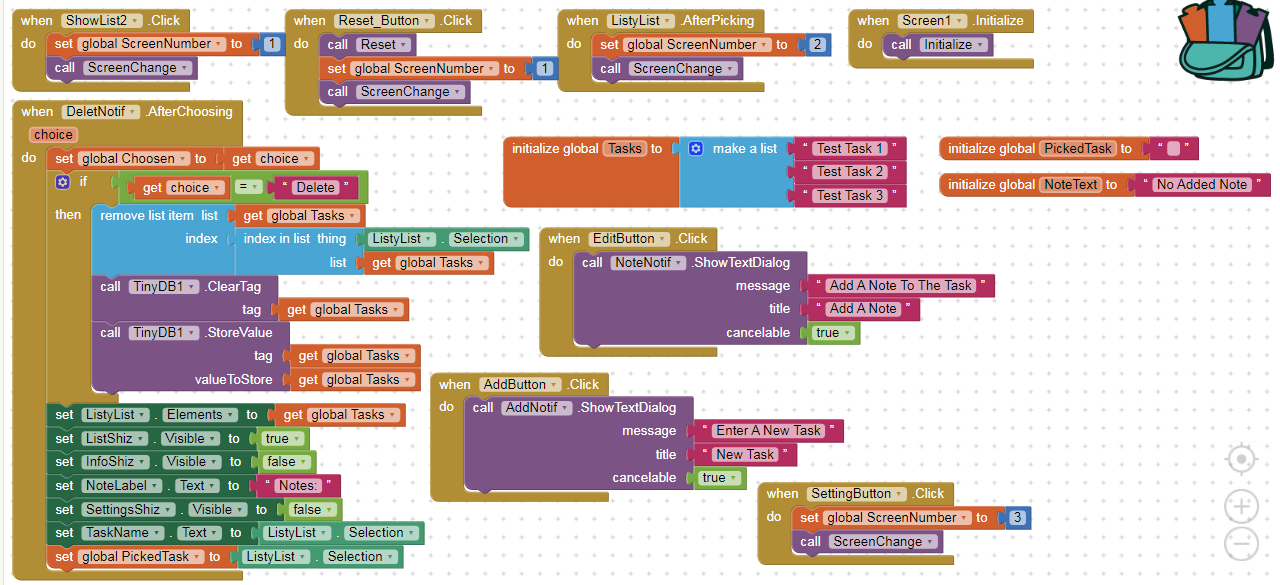
**Write your 2d response here (< 200 words)**

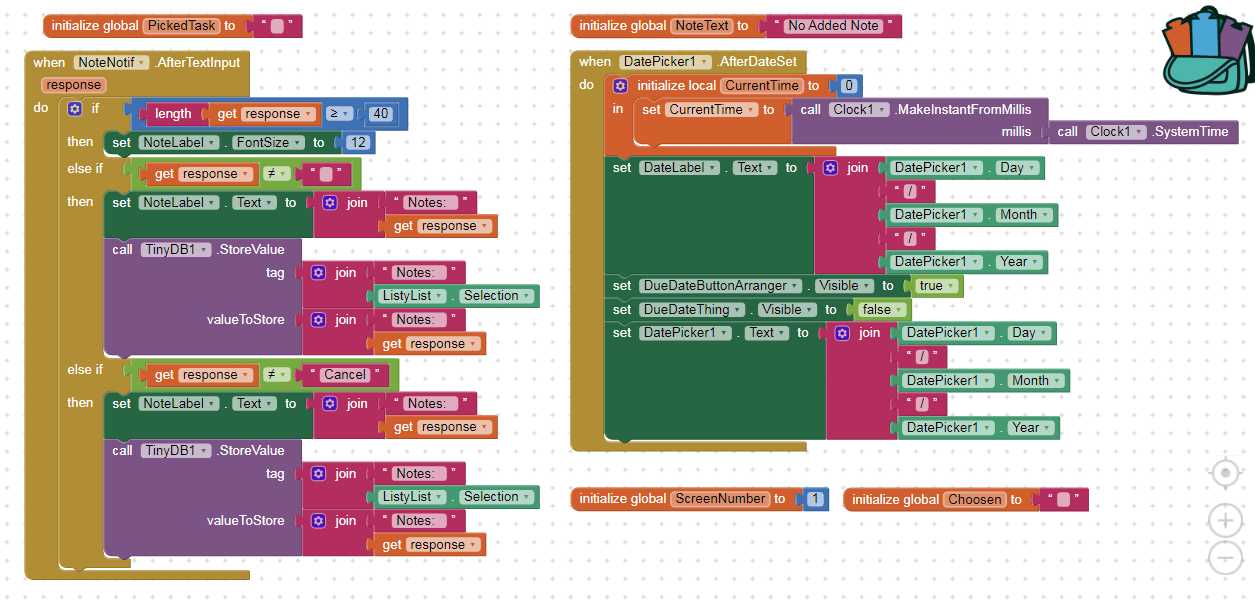
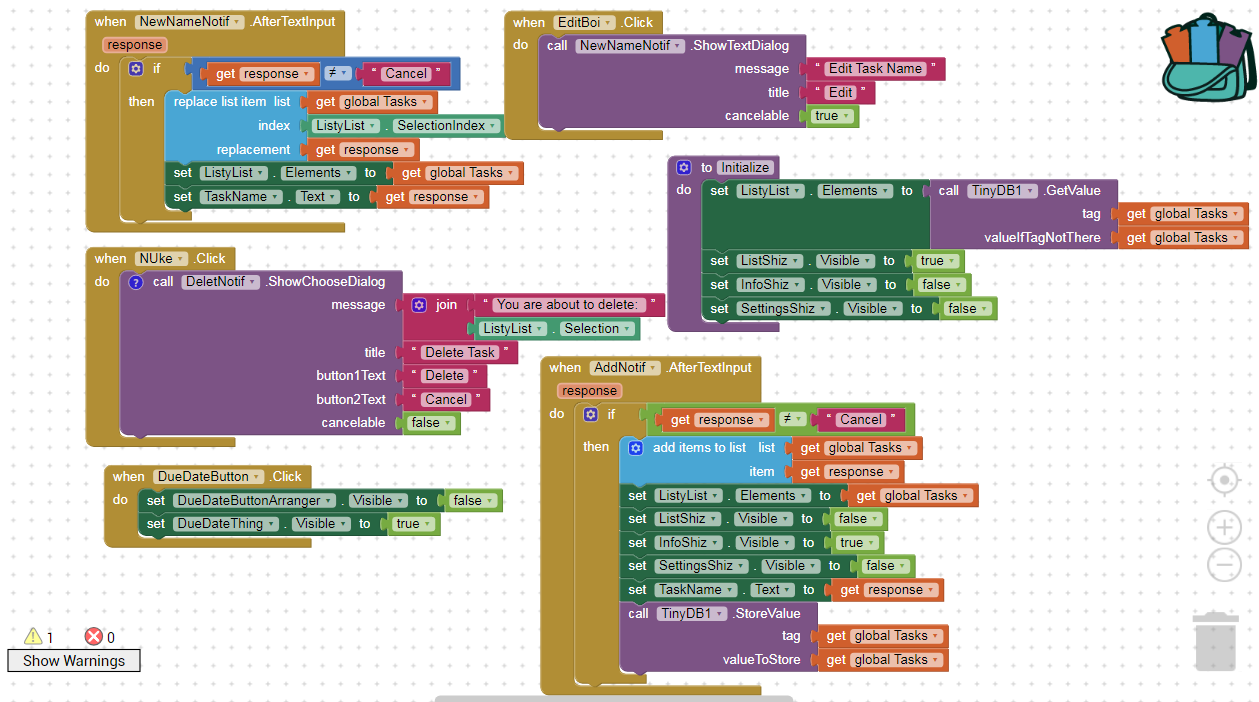
|  |
| --- |
| The code I have below manages the complexity of my program by logically choosing the size of text and easily storing variables in a storage system for the user. This system allows me to put multiple lines of code that would normally be spread out across the program into one section of code that is all in one place. The first else-if statement could have been put into other parts of the code but if the code was put into other places it would make the program cluttered and the section of code would have to be repeated many times. Since I put the code at this point in the program it runs after all the other places the code in this algorithm would run if they were separated and repeated many times. This allows me to put condense my code and make it less complex but with the same amount of functionality with less room for error. |





****

****

****