Project Documentation: Budgetz

Team: “It’s actually called B-” Written by Zoe Roth

# **General**

The group “It’s actually called B minus” created Budgetz as a simple app to get college students to keep track of their financial accounts one week at a time. College students are out in the real world on their own and need some help to keep track of how much they can spend on food, school book or entertainment. Our app has simple and intuitive because budgeting should be the easy part of an engineering or premed student’s day.

# **Marketing**

After going through many different possible app names such as Balanced Budgets for Bougie Teens or 8UD63T$, it was decided that something easy to say and remember works best in the app market. Our research found that many popular budgeting apps connect to bank accounts, are not free, and have many busy screens. Young adults, from the internet age, are wary of giving away bank account info, especially to an app they cannot try yet. Broke college students need to save every cent for food and loans so buying a budgeting app is counter intuitive. A visually busy app with too many screens or choices can be overwhelming and difficult to navigate resulting in a teenagers deleting the app almost immediately. Using all of this information, our free Budgtz app contains four cleanly designed screens and simple input buttons to be an effective and well liked product. The app is for a week because plans change constantly for busy students. A bonus feature is built-in normalizing of categories so that the user does not have to worry about adding all the categories to be equal to 100 percent.

# **Screen 1: Welcome**

## The app starts on the ‘Welcome’ screen with 2 buttons asking to either create a new budget or to view a previously created budget. The backend takes the input and the button will either take the user to page 2 (enterBudget) or page 4 (main). See Figure 1 below to see the app’s GUI for the opening page. The code for Welcome.java allows the user to change screens as seen in Figures 7a and 7b.

# **Screen 2: Enter Budget**

## If the ‘Edit/New’ button is pressed, the user is sent to page 2. This page has a text box to input the money allotted to be used in the upcoming week. When the ‘Enter’ button is pressed, the value is saved and will be carried to the next pages by using “shared preferences” in the code. The User gets sent directly to the third screen. Figure 2 is the GUI for our ‘Enter Budget’ screen 2 and Figure 8 is the backend and interface code found in enterBudget.java .

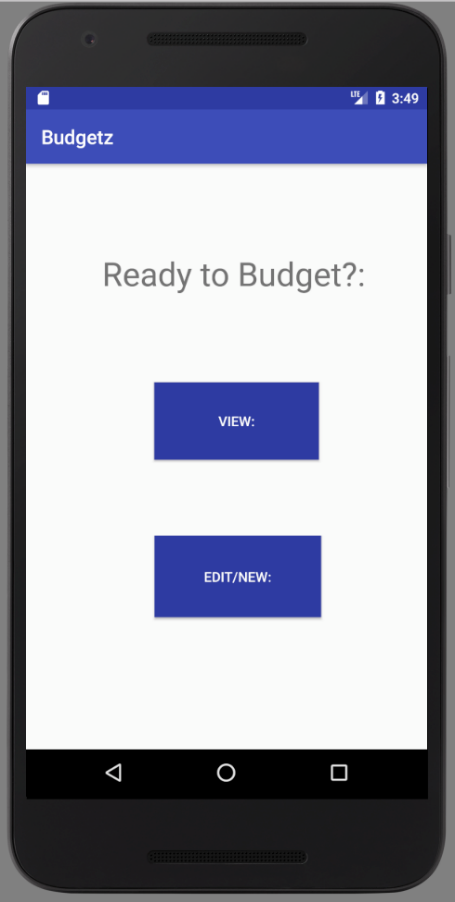
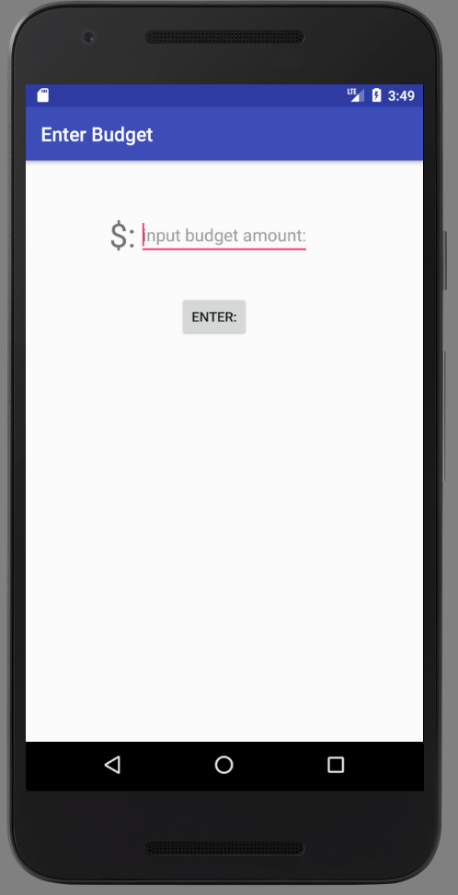
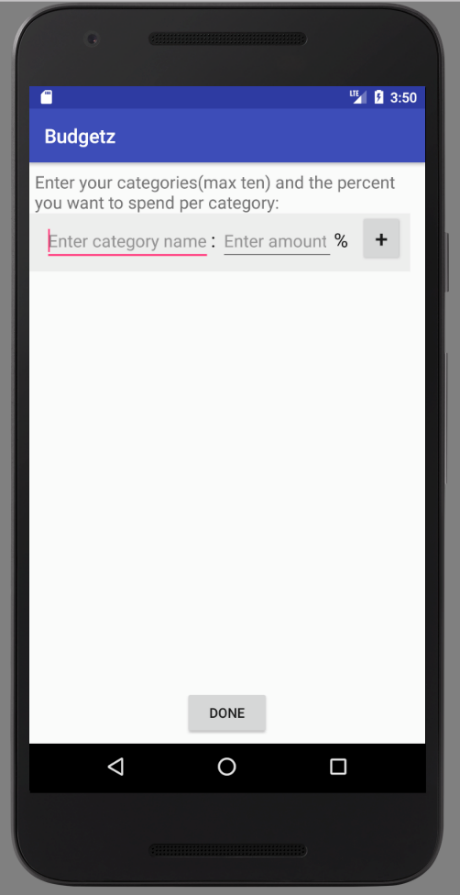
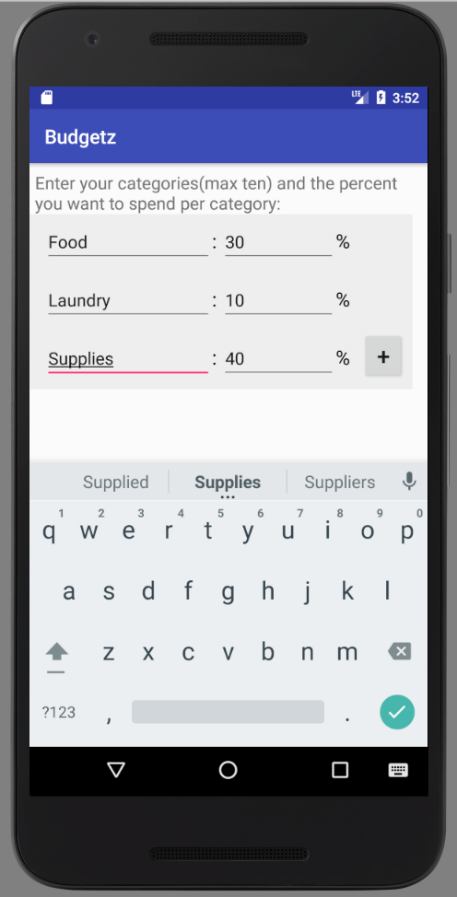
# **Screen 3: Set Up Budget**

## Screen 3 opens with a text box for the category name, percentage of budget and a button to add a category. At the bottom of the screen there is a save button to continue on to the final screen after the all the wanted categories are entered. The category names and percentages are saved in arrays of strings and passed on by using “shared preferences”. In the backend of the code, there is a max of 10 categories hidden from the user which become visible when the plus button is pressed. Also, before the information is passed through, the categories are normalized. If the category percentages don’t add up to 100 percent exactly, calculations are done to fix this user error. Figure 3 is an example of the screen when it opens to a blank category. Figure 4 is an example of user inputs. Figures 9 a-e are Categories.java where the names and values are coded to be shared with the rest of the app.

# **Screen 4: Main**

## Main can be accessed with saved values of a new budget being created or will load the arrays created from the last budget created after the last save, even if the app has been closed out of. The GUI was designed to show the categories, the amount of money given to each by the app’s calculations and input boxes that will subtract money spent from a specific category. If a negative value is entered, it will add money back to the category. There is another save button at the bottom of the screen so that the changes are kept even if the app is closed. The only way to lose this information is to create a new budget from the start screen. The backend code has used the preexisting “shared preference” function and kept all of the information in arrays from the start of the app. Figure 5 is our final screen of untouched categories while Figure 6 has spent money from each category and then saved. Finally our backend code comes all together in the final screen which can be seen in Figures 10 a-f, our MainActivity.java.

# **Appendix of Figures**

Figure 1.  Figure 2. Figure 3.  Figure 4.

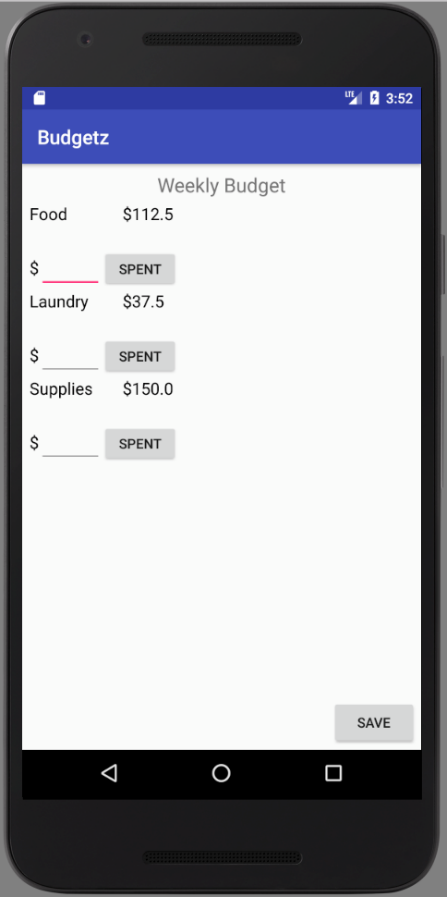
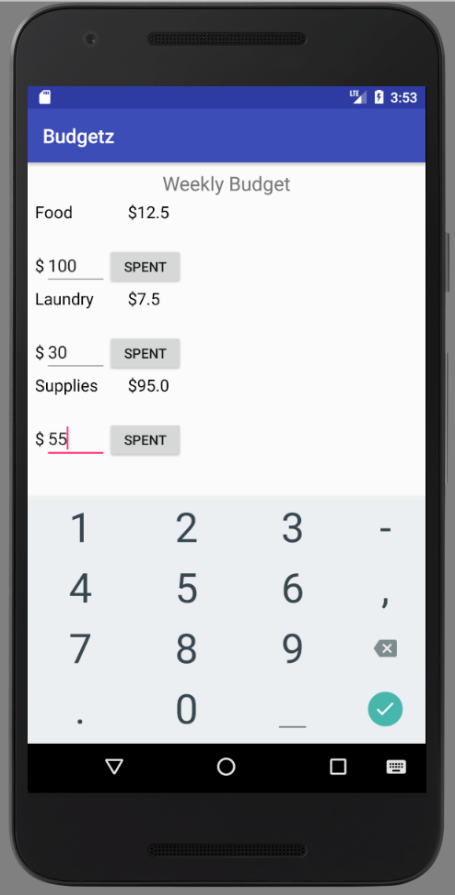
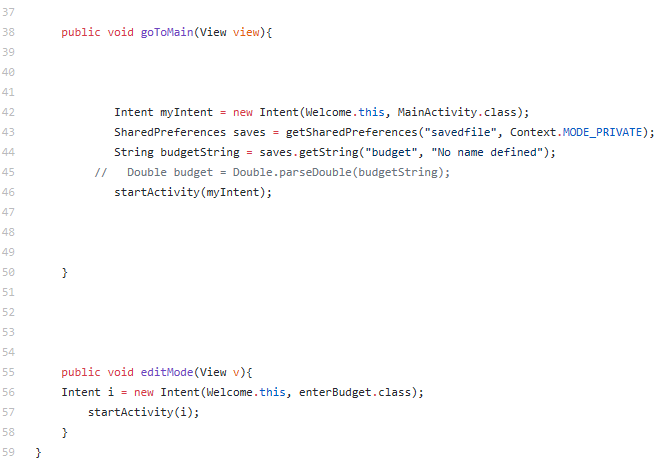
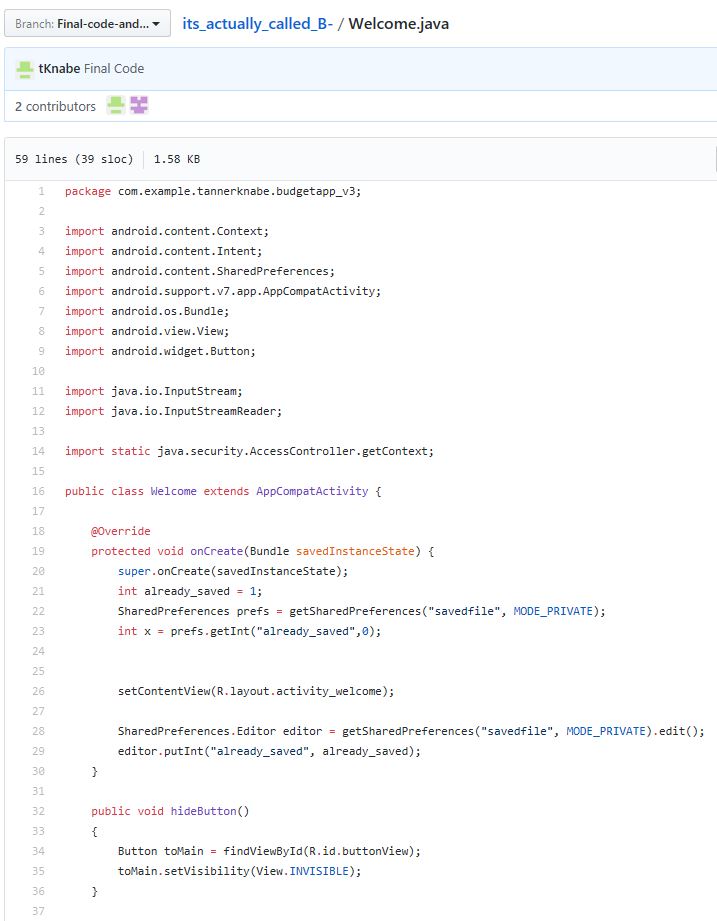
 Figure 5.  Figure 6.

Figure 7a. Figure 7b.



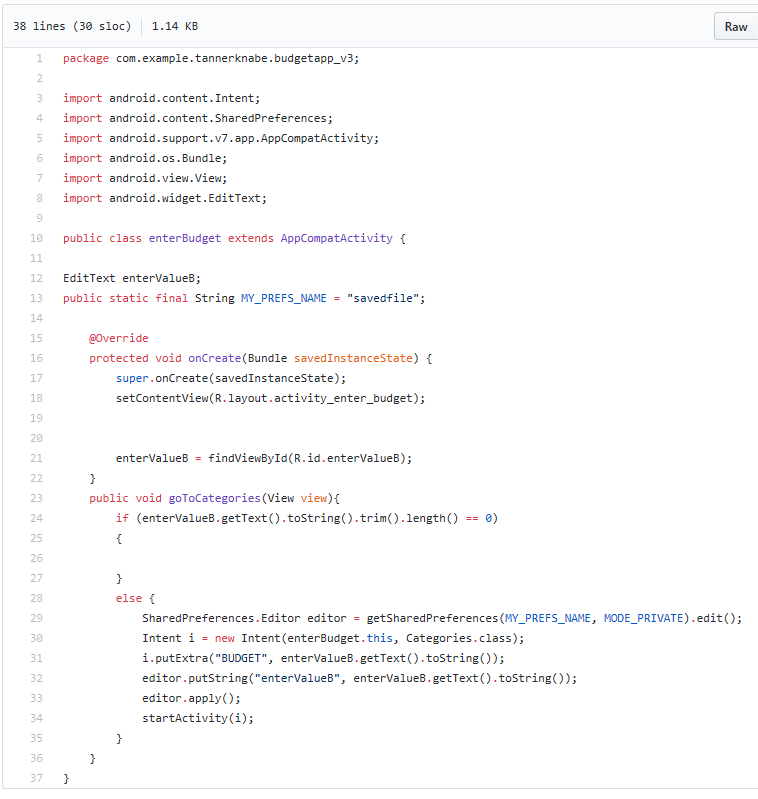
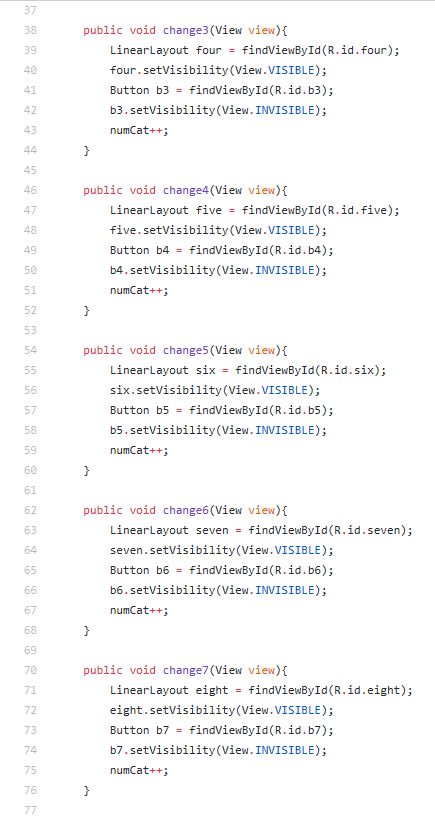
 Figure 8.

Figure 9a.Figure 9b.

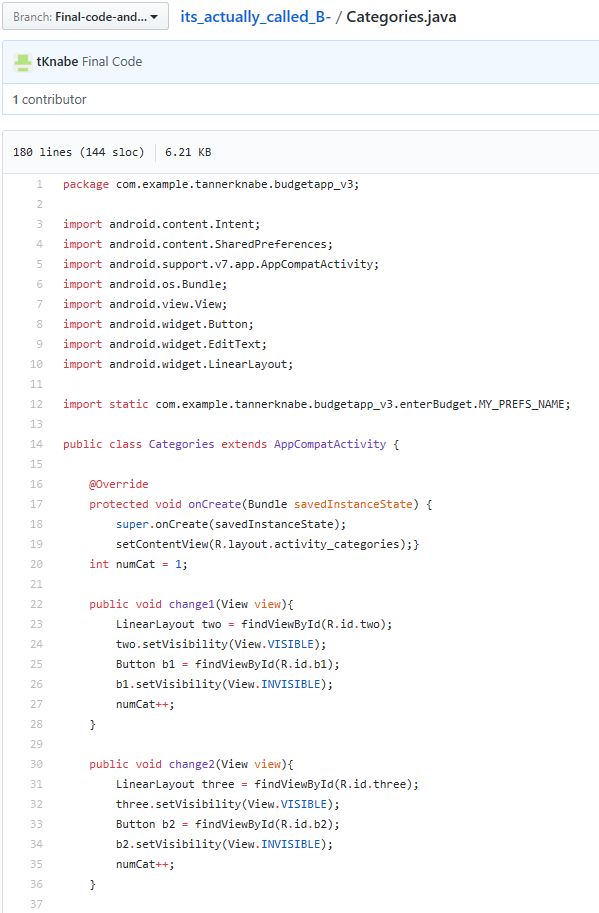


Figure 9c.



Figure 9d.

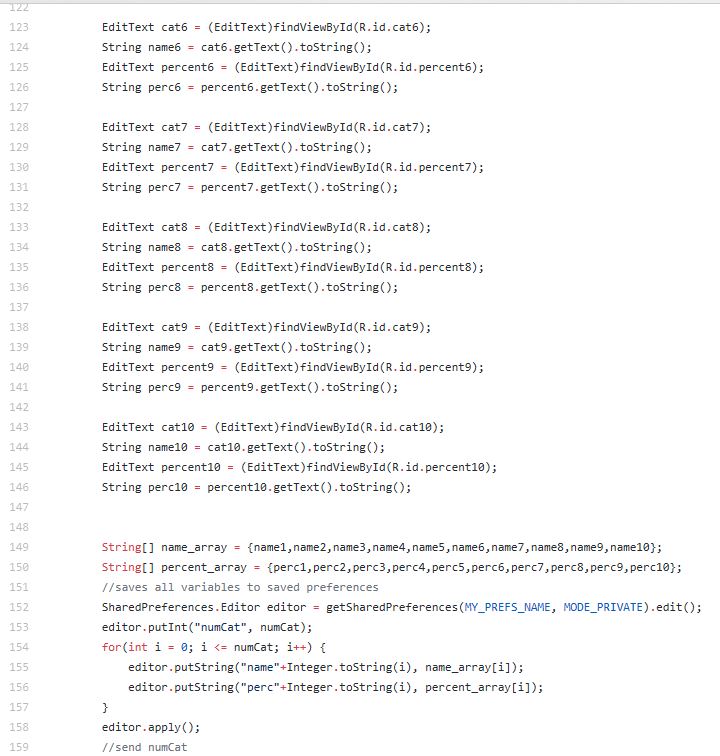


Figure 9e.

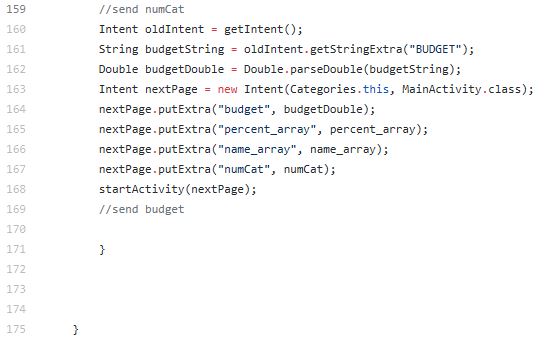


Figure 10a.

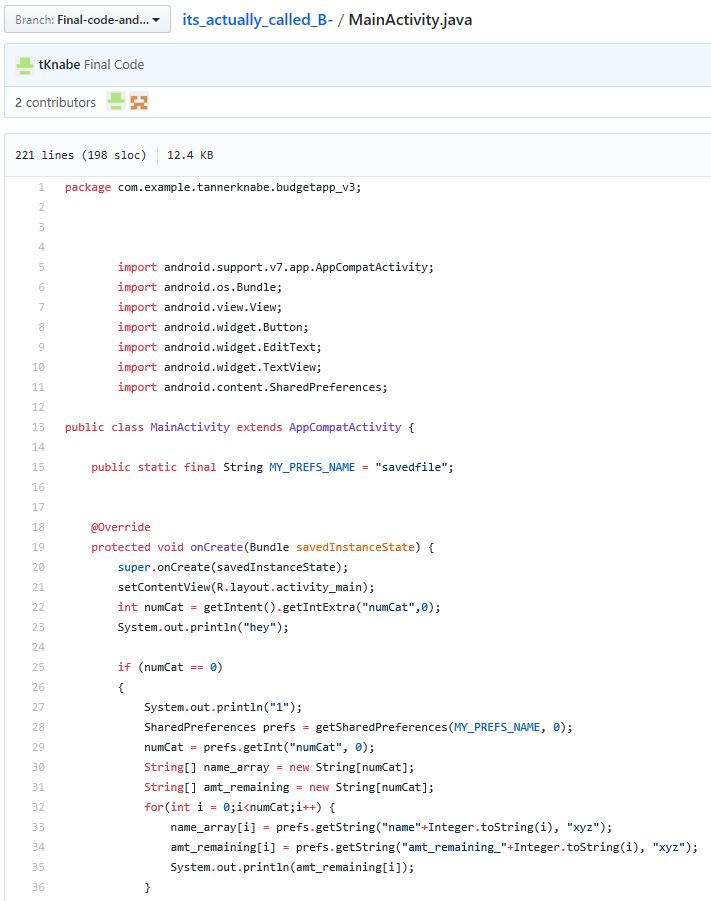


Figure 10b.

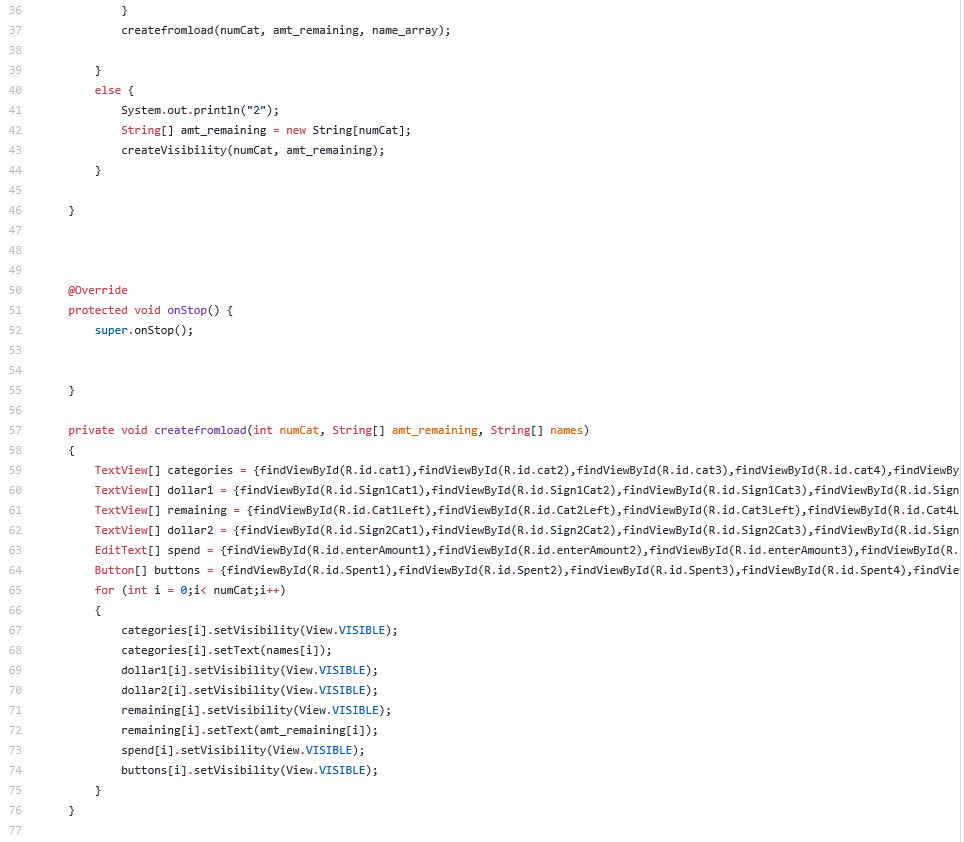


Figure 10c.

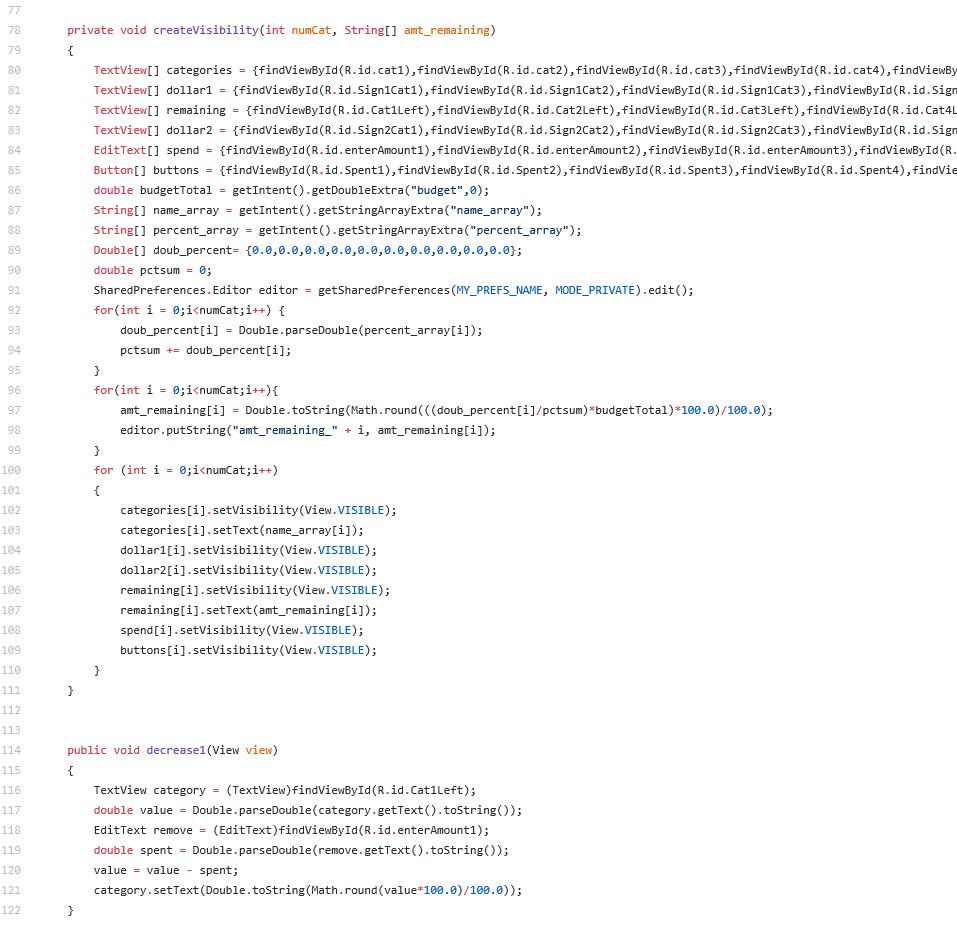


Figure 10d. Figure 10e.Figure 10f.

