

Report assignment 1 - BikeShare V4

Most important design choices, for example, class and layout structures.

In BikeShare V4, the overall design of the program is made of activities. All activity classes do also contain an activity xml-file which interact with each other. Each component in the XML-files are identified with unique id's in order to make it easier for the Java classes to refer to their components (buttons, text views etc.).

The BikeShareActivity-class is the parent activity for all the other activities (EndRideActivity and StartRideActivity). All the rides for bikes can be stored in a list which is identified in BikeShareActivity. This list is used by both StartRideActivity and EndRideActivity to add a bike and its start location and the end location for the ride.

The BikeShareActivity have two `setOnClickListener()` which can navigate to either StartRideActivity or EndRideActivity. The EndRideActivity is only able to store values in the list when the StartRideActivity has stored values in the list at first. All the components in the xml-activities are defined in linear layouts in order to split the page into more parts. As an example, the list of rides on BikeShareActivity is shown in the layout (activity_bike_share/fragment_bike_share) as a list view. All the components of the list View are added in the `getView`-method in the `RideArrayAdapter`-class with unique ids for the bike, start- and end ride. These ids are in the layout defined in another xml-file called `list_item_ride`.

All the more detailed layout is stored in the folder called `values` underneath the `res`-folder. Furthermore, that includes colors for background and texts and sizes of fonts. Each color and font size have got a unique name, which the XML-files refers to when creating a component in the UI.

Short explanation about the user interface of your BikeShare app:

When the app starts to run, the front page is shown. From there you can either choose to add or end a ride by pressing one of the two red buttons. Because there is no added ride when the program starts up, the user should press the "Add ride"-button. Then the user can write a name of the bike in the first text field "What bike?" and in the second text field "Where" the location of the bike. After the text fields are filled out, the user can press the "Add ride"-button. Then the user gets navigated back to the main page. Then the orange list view displays the values which have been added to the list. Afterwards the user can end the ride by pressing the "End ride"-button. From there the user has to make sure that he/she has typed in the exact same name of the added bike to be able to store an ending location of the bike which is typed in the "Ending where?"-text field. At last the user can press the "End ride"-button and the end ride location should be stored to the related bike in the list view.

Extensions compared to BikeShare app V4 (e.g. the challenges from Work Plan #04)

I have tried to implement the fragments that we learned about in lecture 4. But I did not succeed in getting it to work completely. I changed my BikeShareActivity to extend a FragmentActivity instead of Activity which works without problems. I did also set up the BikeShareFragment but it is not used or working at the moment.


How did you test and evaluate your app?

I was mostly using the build in debugger to check that the given values were correct when typed in some input in the given text fields. Sometimes during the progress some of my text fields (Text Views) had one name in one file and another name in another file, where I did not have consistency in the naming of all the variables. That caused NullPointerExceptions and null references. These were easier to spot by debugging and looking at the LogCat-terminal for exceptions and other errors.

When running the app, I used my phone to test the UI and functionality.

Problems:

The main problem for me right now is to implement the fragment in my program fully. I tried to implement the fragment manager and an additional Fragment in my BikeShareActivity, but the app crashes when I try to implement it in the code. My Fragment tries to find the fragment_container-id in fragment_bike_share.xml and checks that if the fragment is null it initializes a BikeShareFragment. But in these currently out commented lines of code (line 27-33 in BikeShareActivity) a NullPointerException happens if I try run it as a part of the other code. It points towards a null-reference which I did not know how to fix (picture below):



```
ity): java.lang.NullPointerException: Attempt to invoke virtual method 'void android.widget.TextView.addTextChangedListener(android.text.TextWatcher)' on a null object reference
```