GitHub Username: bqfix

Dicey Dice

Description

Calling all gamers! Forgot your dice at home? Curious about dice odds? Dicey Dice has you covered, for all your randomization needs! From classic 6-sided dice to d20s and beyond, virtually roll whatever you need for your game, all with a single click!

Have a set of dice you roll constantly? Create custom favorites, with whatever wild numbers you need. Curious about the odds? Check out the detail screen for predicted probabilities, to optimize your decision making!

Intended User

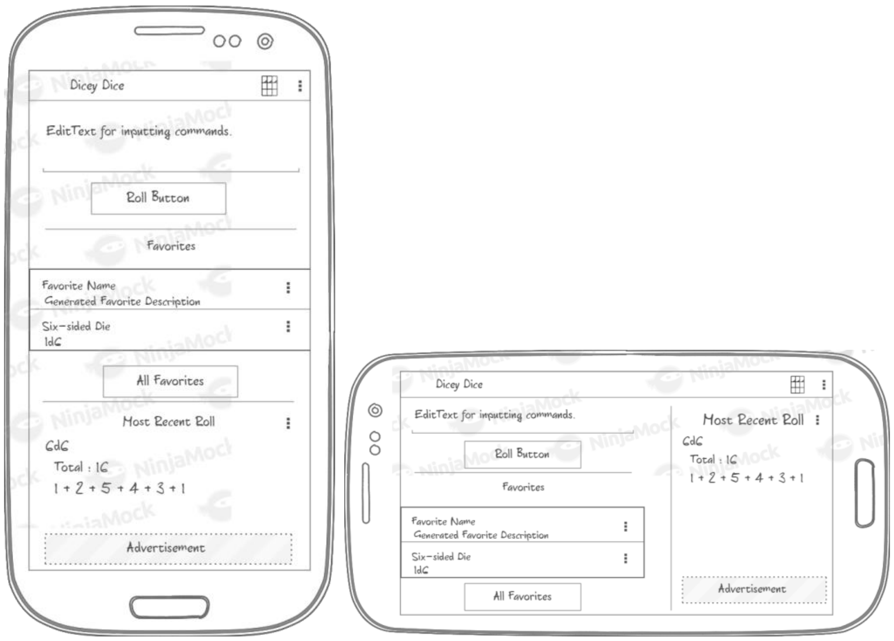
This app is intended for tabletop gamers, both casual and veteran. Simple dice-rolling functionality will be implemented in an easy-to-use manner for people who simply want to see results, while details screens will provide probability odds and other helpful data for power-gamers who want to optimize their gameplay.

Features

- Randomize dice rolls, both preset and custom, and provide the results.
- Save sets of dice (locally or cloud-based) for quick access and rolling.
- Provide a details screen for any given set of dice, displaying probabilities for a given roll.

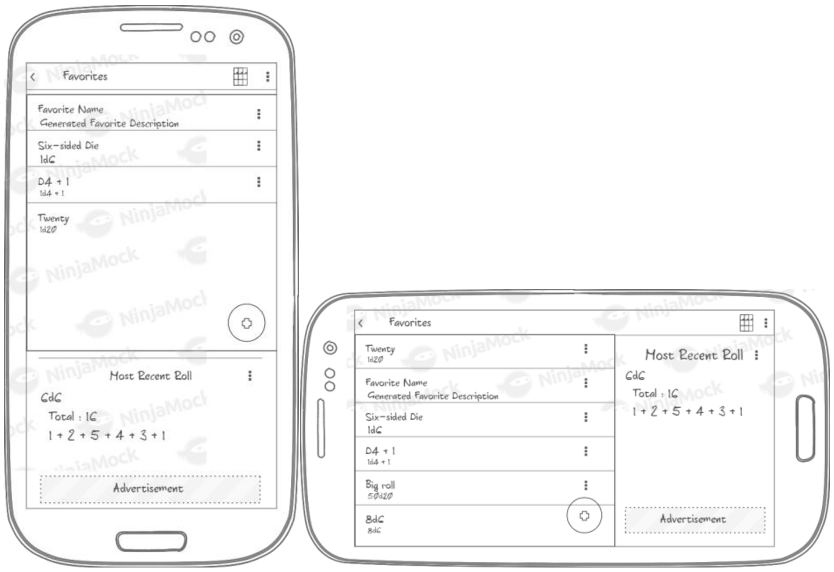
User Interface Mocks

MainActivity



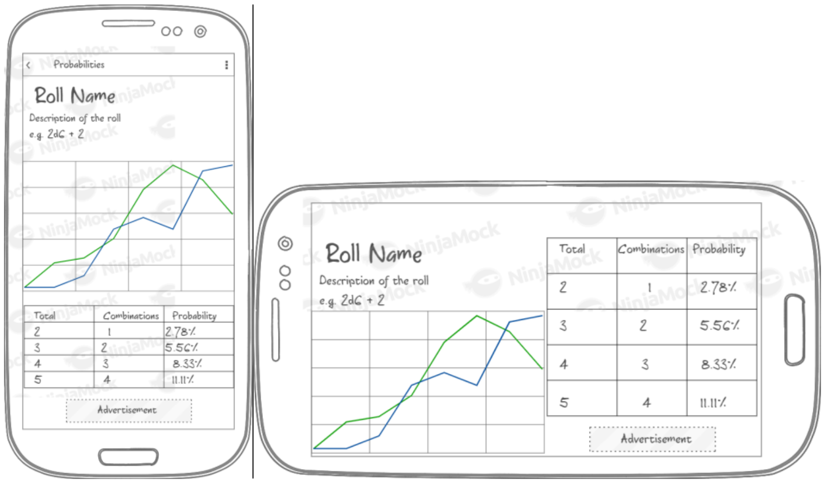
Contains an EditText for inputting and executing roll commands manually, a truncated Favorites list with a button to link to FavoriteActivity, and a display for the results of the most recent roll. Clicking a Favorite rolls the dice and displays the result in the Most Recent Roll section, with the menu button offering options to view Details, Edit, or Delete..
The Most Recent menu allows for saving the roll as a new favorite, or viewing Details
The chart button in the toolbar links to the HistoryActivity.

FavoriteActivity



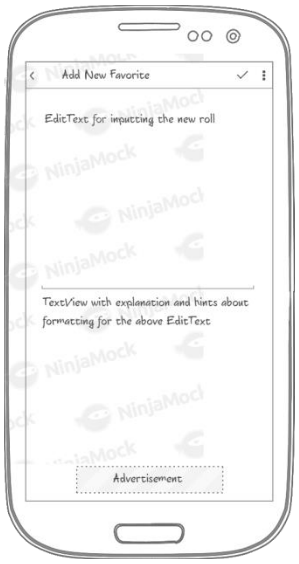
Offers the ability to use any saved Favorite as a roll, simply by clicking on it.
As with the Favorite menu in MainActivity, offers options for Editing via AddFavoriteActivity, Deleting, or viewing Details.
A FAB allows for the addition of brand new favorites via AddFavoriteActivity.
Link to HistoryActivity button continues to exist in the toolbar.

DetailActivity



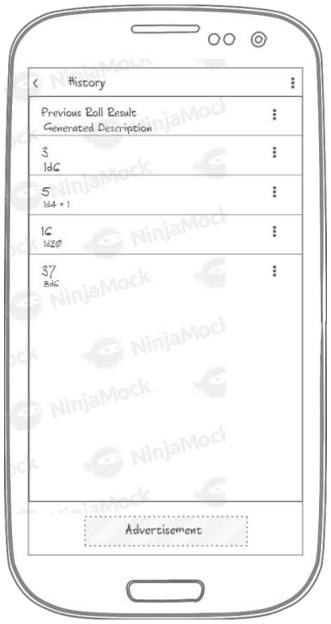
Allows for viewing the details and probabilities of a given set of Dice via graph and a scrollable table.

AddFavoriteActivity


Add New Favorite
EditText for inputting the new roll
TextView with explanation and hints about formatting for the above EditText
Advertisement

Allows for saving new Favorite DiceRolls to the Firebase Realtime Database by EditText (which will be parsed to ensure validity before saving).

HistoryActivity



Simply displays the most recent 100 rolls, accessed from the Firebase Realtime Database.

Key Considerations

How will your app handle data persistence?

Dice sets will be saved to a Firebase realtime database for use across different devices.

Describe any edge or corner cases in the UX.

- An extreme user who wants to save a lot of dice sets to the cloud could eat up space quickly, and there must be per-user limits as such.
- In the event of excessively large dice rolls, system memory can quickly run out and cause crashes during calculations, and so an AsyncTask will be implemented to remove stress from the main UI thread.
- For potential future regional differences/localization purposes, all strings are kept in a strings.xml file, and RTL layout switching will be enabled on all layouts.
- Testing will be performed to ensure accessibility, such as with a D-pad. Additionally, ImageViews will have content descriptions provided for clarity.

Describe any libraries you'll be using and share your reasoning for including them.

FirebaseUI will be used for implementing a login/authentication interface for those users who wish to store dice sets in the cloud.
https://github.com/firebase/FirebaseUI-Android

Describe how you will implement Google Play Services or other external services.

Firebase Realtime Database and Authentication will be used for cloud storage of custom/favorited dice sets.

Firebase Analytics will be implemented for tracking usage of the app.

Google AdMob will be used for populating ads within the app.

Next Steps: Required Tasks

Task 1: Project Setup

- Setup most recent release versions of Gradle, AndroidStudio, and necessary libraries.
- Configure Gradle dependencies for Firebase and AdMob
- Create project
- Link project with Firebase and setup Database Access rules to require authentication
- Configure Firebase Realtime Database
    - Allow about 100 favorites per user.
    - Only save 100 most recent dice rolls.
- Create a custom DiceRoll object to allow for saving and processing favorite dice.
    - Should be created from a delineated string.
    - Need method to parse string and check if valid.
    - How many faces per die, what number per face?
    - How many dice of each type? (1d8, 2d6, etc.)
    - Modifiers (+1 to final roll, etc.)

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Main Activity to allow for inputting dice rolls
    - EditText for inputting rolls
- Build UI for Activity that shows roll details and probabilities.
    - Initially displays probablities in a TextView
- Build UI for Activity to display favorites
- Build UI for Activity to display history of rolls (most recent 100)
- Build UI for Fragment to display Results
    - “Most Recent Roll” to be displayed at the bottom of Favorite and Main Activities
- Build UI for Activity to save custom dice rolls

Task 3: Implement Utils class

- Methods to be used amongst multiple classes.
    - Display of most recent roll, etc.
    - AsyncTask to return DiceRoll result.

Task 4: Implement Dice randomization logic

- Create algorithms to parse a given DiceRoll object.
- Return random result from DiceRoll object.
- Also allow for returning the probabilities of a DiceRoll object.

Task 5: Implement Authentication

- Use FirebaseUI to create an authentication screen allowing for e-mail or Google login.
- Disallow use of Main Activity unless the user is logged in.
- Ensure that the back button exits the app when pressed on login screen.

Task 6: Implement Firebase

- Link Activity to save custom dice rolls to Firebase Realtime Database.
- Link History Activity to retrieve most recent 100 rolls from Firebase Realtime Database.
- Add AdMob logic to populate test ads.
- Ensure Analytics data is working.

Task 7: Widget Work

- Create a widget layout to display the favorites list.
- Selecting a favorite will launch the FavoriteActivity and cause a roll using that favorite.

Task 8: Visual Polish and Testing

- Add animations when transitioning between activities.
    - Results/Favorite > Details.
- Ensure consistent color/font patterns.
- Ensure RTL switching capabilities.
- Test to ensure D-pad navigation capabilities.
- Implement Java library to allow for drawing/transformation of polygons to represent dice. (If time allows)
- Alter detail activity to additionally show graph/chart representation of probabilities as opposed to exclusively text. (If time allows)