REPORT

LAB 03

Student name: Nguyễn Đăng Nhã

Student ID: 21IT033

Student email: nhand.21it@vku.udn.vn

Github: https://github.com/dangnha/Mobile-Multiplatform/tree/master/Lab3/dice_flutter

1. Introduction

- Briefly describe the purpose of the lab report:
  - The purpose of the above program is to create a simple dice rolling app using Flutter. It displays two dice images on the screen, and when a user taps on either die, both dice faces are randomly changed, simulating the act of rolling dice.
- Provide background information on your mobile app:
  - Programming Language: The app is written in Dart, which is the programming language used for Flutter app development.
  - Framework: Flutter is used to develop the app. Flutter is an open-source UI software development kit created by Google for building natively compiled applications for mobile, web, and desktop from a single codebase.

2. Objectives

- State the objectives of the lab:
  - Create a simple dice rolling app using Flutter.
  - Simulate Dice Rolling: Implement functionality to simulate the act of rolling dice when a user interacts with the app.
  - Randomize Dice Faces: Generate random numbers to represent the faces of the dice each time a user taps on them.
  - Update UI Dynamically: Use Flutter's state management to update the UI dynamically whenever the dice faces change as a result of user interaction.

3. Methodology

- Describe the methodology used in the lab:
  - UI Design: The program starts by designing the user interface using Flutter's widget-based approach. This includes defining the overall layout structure using MaterialApp, Scaffold, and AppBar widgets. Additionally, Image widgets are used to display the dice faces.

    - State Management: As the program requires dynamic updates to the dice faces, StatefulWidget is employed to manage the state of the application. This allows the program to re-render the UI when the state changes using the setState() method.
    - Randomization: To simulate dice rolling, Dart's Random class is utilized to generate random numbers representing the faces of the dice. These random numbers are generated each time the user interacts with the app.
    - User Interaction: User interaction is facilitated through the use of TextButton widgets. These buttons are placed over the dice images and configured with onPressed() callbacks to trigger the dice rolling simulation when tapped.
    - Feedback Mechanism: Upon user interaction, the program provides visual feedback by updating the dice images with the newly randomized faces. This feedback mechanism ensures that users are aware of the result of their action.
- Explain how your app was developed:
    - Project Setup: The development environment for Flutter was set up, including installing Flutter SDK and setting up an IDE like Visual Studio Code.
    - Project Initialization: A new Flutter project was initialized using the flutter create command or through the IDE's built-in project creation tools.
    - UI Design: The user interface (UI) for the app was designed using Flutter's widget-based approach. This involved defining the layout structure using MaterialApp, Scaffold, AppBar, and Image widgets to display the dice faces.
    - State Management: StatefulWidget was used to manage the state of the application. This allowed the program to re-render the UI whenever the state changed using the setState() method. State variables were defined to represent the faces of the left and right dice.
    - Dice Rolling Logic: A method called changeDiceFace() was implemented to simulate dice rolling. This method generated random numbers representing the faces of the dice using Dart's Random class and updated the state accordingly.

4. Results

- Present the results of the lab.
    - Create a simple dice rolling app using Flutter. It displays two dice images on the screen, and when a user taps on either die, both dice faces are randomly changed, simulating the act of rolling dice.
- Include screenshots of the app.

5. Discussion

- Discuss the results obtained.
    - Functionality: The program successfully achieves its primary objective of simulating dice rolling. Users can interact with the app by tapping on the dice images, triggering the rolling simulation, and seeing the dice faces change randomly.
    - User Experience: The app provides a simple and intuitive user experience. The UI layout is clean and visually appealing, with clear feedback provided to users when they interact with the dice. This enhances user engagement and enjoyment while using the app.
    - Randomization: The program effectively utilizes Dart's Random class to generate random numbers representing the faces of the dice. This ensures that the results of the dice rolls are unpredictable and realistic, providing an authentic dice rolling experience.
- Analyze the strengths and weaknesses of cross-platform mobile app development:
    - Strengths:
        - Code Reusability: One of the most significant advantages of cross-platform development is the ability to reuse code across multiple platforms. Developers can write code once and deploy it on both iOS and Android platforms, saving time and effort.
        - Consistent User Experience: Cross-platform frameworks often offer tools and components that help maintain a consistent user experience across different platforms. This ensures that the app looks and behaves similarly on iOS and Android devices.
    - Weaknesses:
        - Dependency on Frameworks: Cross-platform development relies heavily on frameworks and third-party libraries. Developers may face challenges if the framework lacks support for certain features or if there are compatibility issues with new platform updates.

6. Conclusion

- Summarize the main findings of the lab:

    - State Management: The use of StatefulWidget and setState() method enables dynamic updates to the UI based on changes in the app's state. This facilitates the real-time updating of the dice faces when users interact with the app.
    - Randomization: Dart's Random class is utilized to generate random numbers representing the faces of the dice. This ensures that the results of the dice rolls are unpredictable and realistic, enhancing the authenticity of the simulation.
    - User Experience: The app offers a simple and intuitive user experience, with clear feedback provided to users when they interact with the dice. This enhances user engagement and enjoyment while using the app.
- Provide recommendations for future work:
    - Additional Features: Consider adding additional features to make the app more engaging, such as sound effects for dice rolling, customizable dice colors or patterns, or the ability to roll multiple dice at once.
    - Localization: Support multiple languages to make the app accessible to a broader audience. Implement localization features to provide translations for different languages and cultures.
    - Animations: Introduce animations to enhance the visual appeal of the app, such as animated dice rolling or transitions between dice faces. This can make the app more dynamic and immersive for users.
    - Settings Menu: Create a settings menu where users can customize aspects of the app, such as the number of sides on the dice, the speed of the dice rolling animation, or the presence of sound effects.