Readme

ANGRY BIRD PROJECT

Students:

- Sidharth Kumar (2023526)
- Rishabh Dwivedi (2023434)

Project Structure

The game consists of several main classes and screens that define the user interface and gameplay experience:

1. **AngryBirds.java (Main Class)**

   Entry point of the game. Initializes the game, sets the initial screen to `StartPage`, and manages the game's primary state. Holds a `SpriteBatch`, which is used to render textures on each screen.

2. **Screens**
   - **HomePage.java**: The introductory screen that briefly displays a splash image or animation, then transitions to the main menu.
   - **StartPage.java**: The main menu where players can start the game or access other options.
   - **LevelSelection.java**: The level selection screen, allowing users to choose from multiple game levels.
   - **Level1.java** and future levels: Represents individual levels of gameplay, where game logic is implemented for user interaction.
   - **WinScreen.java** and **LossScreen.java**: Screens shown based on the outcome of the level.

Flow of Execution

The game's flow begins with initialization in `AngryBirds.java`, where `StartPage` is set as the first screen. The screens are navigated based on user inputs, and each screen has a specific role in guiding the player through the game. Below is a breakdown of each screen's role and flow:

1. **Game Initialization:**
   - The game starts with `AngryBirds.java`, which initializes `SpriteBatch` and sets `StartPage` as the first screen to display.
2. **Home Page:**
   - Displays a splash image or animation and, after a brief delay, transitions to `GameScreen`.
   - **Key Methods:**
     - `show()`: Loads and displays the splash image.

- `render()`: Maintains the display period, then transitions to `GameScreen`.
- `dispose()`: Releases resources when the screen is no longer active.

3. **Start Screen:**
   - Provides options for the player to start the game or explore additional features.
   - **Key Methods:**
     - `render()`: Checks for user interaction on menu options.
     - `dispose()`: Frees up resources once the screen is exited.
4. **Level Selection:**
   - Displays level options that the player can select.
   - **Key Methods:**
     - `render()`: Handles touch input for level selection and the back button.
     - `dispose()`: Releases resources associated with the home screen.
5. **Level1:**
   - Represents individual levels, handling the gameplay mechanics.
   - **Key Methods:**
     - `render()`: Runs the level's game logic and captures user interactions.
     - `dispose()`: Cleans up resources once the level is exited.

Setup and Execution Instructions

To set up, run, and test the project, follow these steps:

1. **Clone the Repository**

   In your IDE (e.g., IntelliJ IDEA), go to `File > New > Project from Version Control > Git`.

   Enter the repository URL, select the directory to clone to, and click `Clone`.

   Repository URL: https://github.com/SidharthKarnatak/AngryBirdsGame2
2. **Build the Project**
   - Ensure you have LibGDX and Java JDK installed.
   - Configure the project to include LibGDX dependencies.
3. **Run the Project**
   - Set `AngryBirds` as the main class.
   - Run the project through the IDE.
