CSE 201 :- Advanced Programming

WILL HERO GAME

Presented by Group 10

Aamleen Ahmed
2020002

Aayush Kumar
2020008

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Implementations and Functionalities

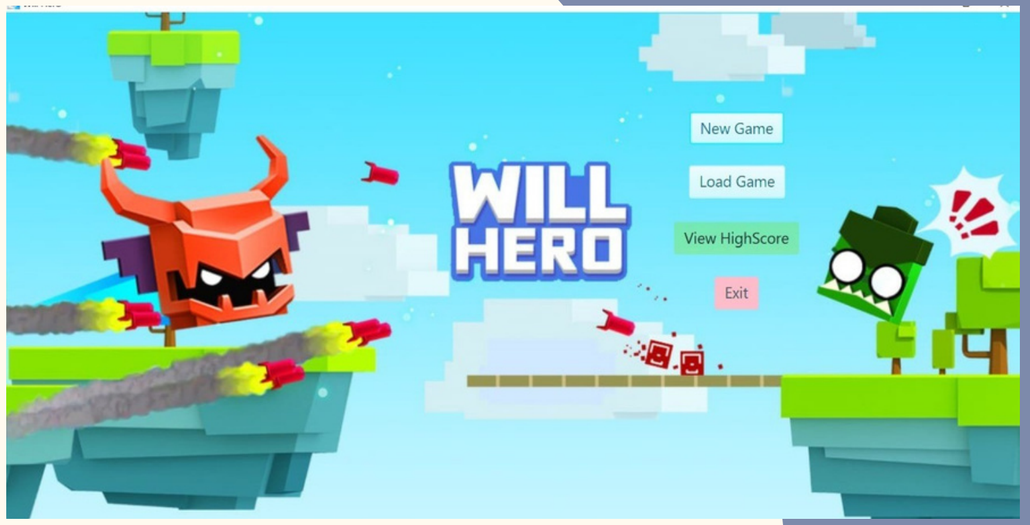
- The objective of this game is to defeat the Boss Orc, along with facing obstacles in between.
- Game class is handling the Welcome Screen, whereas User class is handling the newGame & loadGame. All other inputs are handled by Hero class.
- Whenever the Player collides with a TNT, or comes under an orc, or fall in abyss , the game will stop there and the option to resurrect will be given (if possible).
- Whenever the Player collides with a chest , the player will be rewarded with something. It would be either with Coins or with some Weapon.
- There are different menus on Winning and Losing. When a player lose at some point , a resume menu will pop up and when the player wins the game , a Winning message pop-up.
- Navigation between Scenes is done using FXML Files & FXML Loader.
- While making this game, the basic ideas of Object Oriented Programming were followed strictly, with the game divided in many interfaces, abstract class & normal class, & their relationships being preserved as per the given UML Diagram.
- **Problems Faced:** Synchronizing multiple scenes with Game TimeLine & detecting bugs.

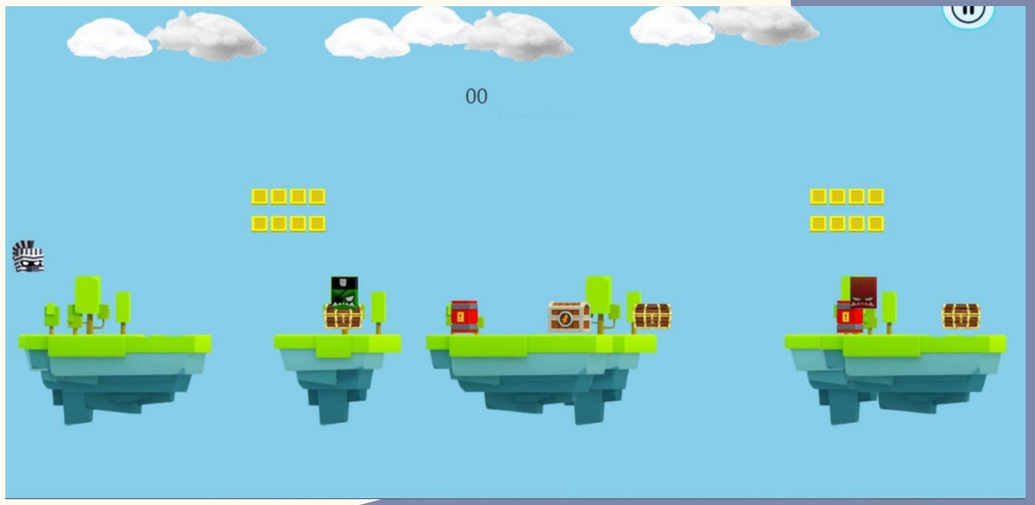
Design Patterns & Multi Threading

Design Patterns:-

- **Observer :-** Multiple event-Handlers are regularly observing for any change in action. Also used ChangeListener() to look into the changes when selecting a menu from Listview.
- **Facade :-** To divide the various actions taken on different types of collisions with Orcs (Hero, Helmet)
- **Template :-** A fixed recipe to create the platform & dynamically place various game objects
- **State :-** To check collisionType between Orcs , Hero and other Objects.
- **Proxy :-** To give access to only the required type of saved game. If not present in database, access denied.
- **Iterator :-** In multiple classes, to iterate over the many game-objects & weapons.

UI :- Using JavaFx Libraries
Images Sources:- Internet and Provided Images
Multi-Threading:- All the animations are implemented using AnimationTimer, each of which is a Thread in Java Virtual Machine (JVM).





Individual Contributions

Although we did all the tasks in a combining effort but the tasks we mainly divided are

Aamleen Ahmed

- Serialization and Deserialization
- FXML Works :- StartGame , LoadGame , ExitGame & Linking of FXMLs
- Working of Weapons using Cloanable
- Movement Using Timeline Animation
- Error Handling
- Interactions:- Between Hero and other objects (Chests, Orcs , through weapon , Islands)

Aayush Kumar

- In GUI , All the Buttons and Background
- FXMLs : Game Ended (for Winning & Losing both)
- Initialization of classes and their attributes
- Designing of Layout:- Placement of Orcs , Islands , TNT , Chests etc.
- Effects :- Chest Opening , TNT Blasting
- Presentation