**David Naußed – Spaceshooter Report**

**Pass:**

1. Refactor all literals (“magic values”)
   * Entity related values (Enemies and player) are in the static Gameplay\_Config class
   * Other config values are in the static Config class
   * Both classes are part of the “design” package
2. Refactor all user-facing Strings into resources
   * All strings are in the strings resources file
3. Refactor the Entity hierarchy to minimize duplicated code
   * Entities can inherit from the wrapable, clampable or entity class. Inheriting from those classes ensures that the specific behaviour gets executed as long as the super.update() is getting called.
   * All enemies (Hunter and Ufo) Inherit from the enemy class which is using the Bitmap component
4. Improve the Star entity class
   * On the star creation, one random value gets saved to modify the star’s scale and speed. High random value -> big scale and higher speed; Low random value -> small scale and slow speed
   * The colour gets randomised through three random int calls and using a full byte (255) as a bound
5. Make a new Enemy type (= a new class!) with a different **behaviour**
   * Hunter enemy which can shoot and looks differently. Also inherits from enemy
6. Implement recovery frames (invulnerability for a short period after taking damage)
   * Recovery frames for a period of time and blinking. XORs the colour
7. Move UI / HUD rendering from Game into a separate class
   * GUI Manager and a different GUI for each game state: Game Over and Gameplay
8. Add sound effects to all important interactions
   * Start sound, collision sound, death sound, get hit sound, pickup powerup sound

**Pass with Distinction:**

1. Give all enemies movement patterns
   * After a certain period of time the enemies walk forward. Similar to space invaders
2. Add player projectiles
   * Player can shoot up to a certain amount of bullets, then the player reloads for a specific time and after keeps shooting
   * Same for the enemies
3. Make use of the SurfaceHolder callbacks to only start the game after the Surface is ready to be used, and to re-initialize the SurfaceView when it changes.
   * Implemented the surfaceCreate lambda and surfaceDestroyed which modifies the boolean “m\_initialised” -> if initialise, update, if not don’t update
4. ~~Convert the game to a~~ [~~Flappy Bird~~](http://flappybird.io/)~~-clone with (randomly generated) pipe obstacles instead of enemy ships, and background graphics instead of stars~~.
   * Didn’t do that one!
5. ~~Save and display a high score~~ **~~list~~** ~~(eg. Multiple values), instead of a single high score.~~
   * Didn’t implement this one either
6. Add power ups to the game!
   * Dual\_Gun -> Higher fire rate and two blue bullets instead of one green bullet
   * Health up, on collision raised health

**Assets:**

Art: <https://www.kenney.nl/assets/space-shooter-redux>  
Font: <https://www.fontspace.com/darrell-flood>  
Sounds: <https://beepyeah.itch.io/8-bit-sfx-pack> and by myself