

# Documentation

## Factory Outbreak CG UE 2020S

Xyruz Kyle Eleazar | 11778277

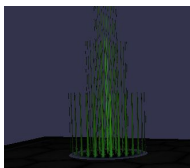
### Gameplay Description

Aphelios (Player Character) is able to walk and jump on surfaces and platforms. There are various collectable objects spread throughout the levels, which improve Aphelios' ability to walk faster, jump higher and etc. Obstacles and Traps have to be avoided, as they can slow him down or even incapacitate him, failing the level and having to restart. (NOTE: After quitting or winning, the game will pause and the console will prompt to enter. INTENDED!)

#### Collectibles & Activatables & Traps



Health Pack    Increases health by 1



Checkpoint Beacon    Checkpoint to finish the level

#### Obstacles & Traps



Trap    Reduces health by 1  
Slows by 50% for 2 seconds  
Disables Sprinting

### Controls

Character		Game	
W	Run Forward	Esc	Close Game
S	Run Backward	F1	Toggle Wireframe Mode
A	Strafe Left	F2	Toggle Culling
D	Strafe Right	F3	Toggle Camera (Player/Debug)
Space	Jump	F4	Toggle Cel Shading
Shift (Hold)	Sprint	F5	Toggle Contour (Cel Shading)

## Camera (Debug)

W	Run Forward	D	Strafe Right
S	Run Backward	R	Strafe Up
A	Strafe Left	F	Strafe Down

## Features & Implementation

### Gameplay

- **3D Geometry**

All visible 3D **GameObjects** are imported via the **OBJLoader** [1] (Player, Platforms, Health Pack, Checkpoint, Trap, ...) and consists of the **Model**, which consists of **Meshes**, which consists of **Vertices**.

- **Playable**

Game can be run without errors. Player can be moved with **keyboard callbacks** and the view can be rotated with the **mouse input**.

- **Advanced Gameplay**

The player has to navigate through the level without falling and without letting its health reach 0. The faster the player gets to the checkpoint, the higher the points will be awarded to the player. After the player falls or depletes all their health, the level will be restarted and the player will be put back to the starting location.

- **Min. 60 FPS and Framerate Independence**

All gameobjects (including player and camera) are moving and rotating independent of the framerate (with DeltaTime / dt).

- **Win/Lose Condition**

- Win Condition: Reach the Checkpoint
- Lose Conditions: Player falls off the platform OR Player's health falls below 1

- **Intuitive Controls** [2]

The player is controlled with the standard key set (WASD & Space) and the mouse and is viewed from a third-person-perspective.

- **Intuitive Camera** [3]

The camera is bound to the player and follows it's moves. There is also a Debug camera, which is not bound to the player (F3).

- **Illumination Model** [4]

The main light source follows the camera and diminishes in the scene with distance. Health packs, traps and the checkpoint have a light source as well. Each object has a standard material, which has parameters to change the final surface, as well as normal vectors.

- **Textures**

Each object has its own texture (image) attached to it, which is properly attached via its texture coordinates (UV). The images are loaded with the help of a library: SOIL2 (Simple OpenGL Image Library 2) [5].

- **Moving Objects**

Most of the objects in the game are moving in one way or another. The player can be moved with keyboard and mouse inputs, some platforms are moving back and forth (done via sinus-offset), the collectible health pack(s) as well as the checkpoint objects are rotating around its own axis.

- **Documentation**

This document.

- **Adjustable Parameters**

- Window:
  - Width, Height, Title, Fullscreen, Resizable
- Graphics
  - Refresh Rate, Sample Rate, Brightness
- Camera
  - Field of View, NearPlane, FarPlane, Sensitivity (Rotation)
- Keyboard Bindings
  - Player: Forward, Backward, Strafe Left, Strafe Right
  - Camera: Up, Down
- World
  - Force of Gravity

- **Collision Detection** (Basic) [6]

The player is able to walk and jump on platforms and is able to collide with certain objects, such as the Health Pack, Trap and the Checkpoint.

## Effects

- **CPU Particle System** [7]

A generic particle system is implemented and can be seen on the checkpoint object. The particle system takes in a template object, which will then be instantiated as many times as the maximum number of particles set. The lifetime of each particle is randomly set after the particle has reached the end of their lifetime.

- **Hierarchical Animation**

Implemented on two objects:

- Health Pack: Two spheres are orbiting around the Health Pack
- Player: Two different spheres start orbiting around the player upon collecting a health pack

- **Specular Mapping**

Implemented for the standard Shader (*SHADER\_BASE*) in the fragmentShader file. (Currently, only the platform has a proper specular map image)

- **Cel Shading** [8]

Simple Cel Shading is implemented and can be toggled with F4. The Implementation details can be found in the fragmentShader file.

- **Contours via Backfaces**

Implemented as described in the Revision Course 2019 [9]. Can only be seen when CelShading is turned on. Can be toggled with F5.

## Additional Libraries

### INIReader

- For reading INI-Files (used for settings.ini game configuration)
- Source: <https://github.com/benhoyt/inih>

### SOIL2 - Simple OpenGL Image Library 2

- For reading Image files (used for loading texture images)
- Source: <https://github.com/SpartanJ/soil2>

### GLM - OpenGL Mathematics Library

- For diverse math usages
- Source: <https://glm.g-truc.net/0.9.9/index.html>

## References:

[1] [Custom OBJ File Loader](#)

[2] [Player Movement](#)

[3] [3rd Person Camera](#)

[4] [Lighting](#)

[5] [SOIL2 - Simple OpenGL Image Library 2](#)

[6] [Axis-Aligned Collision Detection](#)

[7] [Particles / Instancing](#)

[8] [Cel Shading Tutorial](#) / [Culling and displaying Back-Face](#)

[9] [Cel Shading and Contours Revision Course 2019](#)

[Transparency in Meshes](#)

[Player Model \(Aphelios from League of Legends\)](#)

[Trap Model \(Yordle Snap Trap from League of Legends\)](#)