Shifty Duels [tbc]

<1.0>

Konstantin Schaper  
David Kuehlmann  
Alexander Hoffmeister

Video Game Design (CGP603)

Unit Lead: Christian Brindley

[Concept Document 4](#_Toc471430735)

[Game Goals 5](#_Toc471430736)

[Story Overview 6](#_Toc471430737)

[Game Controls 7](#_Toc471430738)

[Technological Requirements 8](#_Toc471430739)

[Title/Start Screen 9](#_Toc471430740)

[Game Flowchart 10](#_Toc471430741)

[Loading Screen 11](#_Toc471430742)

[Game Camera(s) 12](#_Toc471430743)

[HUD System 13](#_Toc471430744)

[Player Character(s) 14](#_Toc471430745)

[Player Metrics 15](#_Toc471430746)

[Player Skills 16](#_Toc471430747)

[Combat 17](#_Toc471430748)

[Health 18](#_Toc471430749)

[Scoring 19](#_Toc471430750)

[Universal Game Mechanics 20](#_Toc471430751)

[Game Levels 21](#_Toc471430752)

[Monetization 22](#_Toc471430753)

[Appendix(es) 23](#_Toc471430754)

# Concept Document

**Game Name:** Shifty Duels [tbc]

**Genre:**. Turn-Based 1on1 Arena

**Description:**

Shifty Duels is a turn-based, one versus one arena game that is played on a square grid.

The game either ends by one player dying or when a set number of turns have been played. The players gain points for dealing damage to the opponent, healing themselves or taking secondary objectives. The player with the most points at the end of the game wins.

The matches take place on floating debris in space with varied terrain. At the end of every turn the terrain randomly shifts in height, so the player has to adapt on the fly.

**Game features:**

* 1vs1 matches
* Isometric on a square grid
* Constantly shifting surroundings
* Variety of characters
* Interesting array of abilities for each character
* Different ways of winning the game

**Resources:**

* Unity (+ MonoDevelope/Visual Studio)
* Photoshop
* 3dsMax
* Audacity
* Trello
* GitHub (+ SourceTree)
* MS Office

# Game Goals

Our game is a turn-based multiplayer arena, the goal is to either kill your opponent or to have the most points at the end. The players spawn on opposing sides of the maps. Both have a hero that has several skills at their disposal, clever use of those skills is the key to victory.   
The game will be played on a square grid of tiles and the main feature of the game comes into play at the end of each turn, at this point the height of the individual tiles changes changing the look of the arena.   
As the game starts both player will find their heroes on opposite sides of the arena. Each player will have a turn consisting of moving his character and attacking his opponent if he wants to or is able to.  
To do this he has an array of four different abilities, that can target the environment, himself or the enemy. Use of these abilities consumes resources and they will incur a cooldown.   
After both players had their turn, the height shift comes into play. This mechanic will change the height of the individual tiles of the arena. This may block line of sight need for abilities or attacks or open new ones.   
The game will end when one player kills the opposing players hero, thus winning the game.

The game will be developed in Unity and be playable on PC.

# Story Overview

No direct story involved in the gameplay is planned for the game. The game is solely a multiplayer arena and does not focus on storytelling in any sort of way.  
A short “backstory” of the arena would be the following:

The setting is an abandoned space station that has accumulated a lot of space debris. While no organic life exists here, old robots from the stations golden days still roam the it. These robots fight for resources to maintain their mechanical bodies. Made all the harder by the ever shifting properties of the debris and the haywire leftover controls of the station.

# Game Controls

The game is played on a square grid. Everything can be controlled with the mouse, key bindings for abilities and menu.

Movement is done by clicking the character then choosing move and clicking where you want to move inside the shown move radius.

After that the player can use skills, by either using a hot key or clicking the skill, every skill has a radius where it can be used.

# Technological Requirements

The game will be done in Unity.

It will need at least the following requirements

* Windows XP SP2+
* Graphic card: DX9 or DX11 with feature level 9.3 capabilities
* CPU: SSE2 instruction set support

# Title/Start Screen

**Start screen:**

The start screen will have a number of option these are: Play, Options and Exit

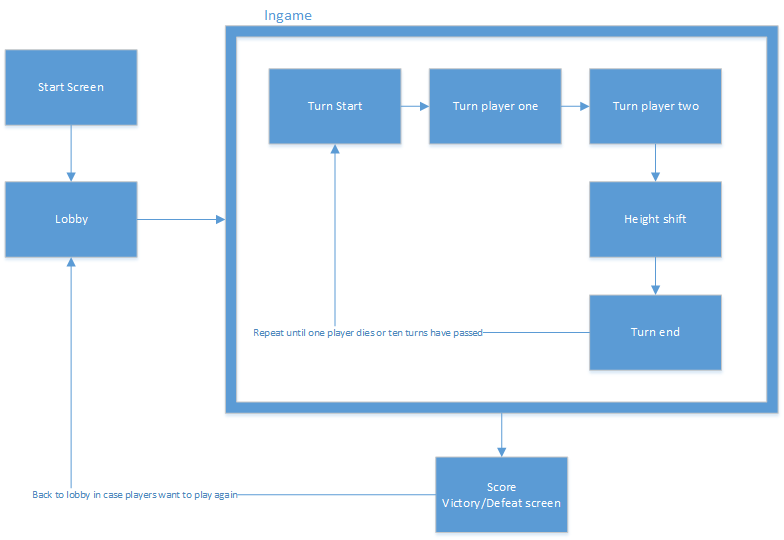
**Options:**

The options provide the possibility to change various game settings such as screen resolution and adjusting audio levels.

**Play screen:**

Here the player can choose his hero and host or join a custom game.

# Game Flowchart



# Loading Screen

The Loading screen will, where it is needed, show an illustration of the arena.

# Game Camera(s)

The camera will be isometric, top down, focused on the map rather than being focused on the players hero. It can be centred on the hero by tapping “SPACE”. It can be moved by using the arrow keys or moving the mouse to the screen borders.

# HUD System

The HUD will contain a number of elements necessary for the game. Since the player control a hero it is imperative that they know the status of their hero at all times. As such we will have displays for health, action points and skills. All of this will be in the bottom centre of the screen.

In addition the player will be provided with a mini map of the arena. It will be located in the bottom right corner.

In the top middle of the screen the score will be shown.

# Player Character(s)

There is no singular player character instead the player can choose a predefined hero at the start of each match.

# Player Metrics

Health, when this stat goes to 0 the hero dies.

Energy, needed to use skills and move around the map. Regenerate each round.

Movement distance, determines how far the hero can move each turn. Movement costs a small amount of energy.

For the prototype hero it would be as follows:

|  |  |
| --- | --- |
| **Health** | 500 |
| **Energy** | 200 |
| **Energy/Round** | 100 |
| **Movement distance** | 5 |
| **Energy cost per tile** | 6 |

# Player Skills

Since we do not have a single player this will be a list for a prototype hero.

Turret Hero [development name]:

Skill 1:

Place a turret, within a defined radius, on the map.

If the turret is already placed move it to a new location instead.

Skill2:

Both the character and the turret shot their weapons at the targeted location.

Skill 3:

Pushes all other characters away from the character and the turret.

# Combat

Combat will mostly be either attacking a target or using a skill. In both cases after choosing the respective action an attack radius will be shown to show where you can attack or use a skill. Confirm simply by clicking the target.

# Health

Health will be tracked as a simple health bar, when falling below a certain percentage a sound effect will be played and the bar will turn from green to yellow.

Should health fall to 0 at any point the hero dies and loses the match.

# Scoring

At the moment the score will be directly derived from damage dealt. If a match goes too long the winner will be the one with the higher score after a set amount of rounds.

# Universal Game Mechanics

The important mechanic for this game is the height shift mechanic after each turn.

An algorithm iterates over the playing field and lowers or elevates all tiles based on the elevation of their neighbouring tiles. This is to prevent player from getting stuck. This mechanic provides a changing field of battle with shifting line of sights and eventually height boni for the heroes.

# Game Levels

The will be one level which will be the arena. It would be a derelict space station and debris.  
The galaxy and a planet as a backdrop. Music should be mainly in the background. Soundeffects need to be clearer, as they underscore the abilities.

# Monetization

The game will be a one time purchase. Further money could be made by selling DLCs containing new maps and heroes.

# Appendix(es)

# Task List

[] = Identifier

Used to refer to a certain part of the project in documents.

[T] = Task

Identifier for tasks.

[TG] = Game Design

All game design related tasks or subtasks fall under this category.

[TGC] = Character Design

In this game, two players each play as a character in a one versus one battle on a lost space station. The characters and their abilities need to be designed. This task revolves around the following subtasks:

[TGC01] = Character Template

Come up with an Microsoft Excel sheet template to later design characters with. It should be generalized and contain all necessary values and fields for the development team to start creating the ingame class from.

The ouput of this task is said excel sheet and is given tot he developer(s) for review. If they are sure they are going to be able to implement the template, this subtask is successfully completed.

[TGC02] = Character Class A

Requires TGC01.

Develop a unique character class built with the required template. This character should be very basic and only use, yet try include all core mechanics.

Output is an excel sheet containing all of the character’s statistics and abilites. The document is given to the developers for review. If they are sure they are going to be able to implement the template, this subtask is successfully completed.

[TGC03] = Character Class B

Requires TGC01.

Develop a second unique character class built with the required template. This character should push the game mechanics to the limit but not be unrealistic to implement.

Output is an excel sheet containing all of the character’s statistics and abilites. The document is given to the developers for review. If they are sure they are going to be able to implement the template, this subtask is successfully completed.

[TGM] = Game Core

All features of the gameplay need to be defined and properly described. This design describes the concept document in detail and extends it.

[TGM01] = Turn Structure

Define all steps each turn of the game consists of.

[TGM02] = Height Shifting

Define the process which changes the terrain every turn.

[TGM03] = Score

Define how score is measured and which actions bring score points.

[TGM04] = Controls

Define the controls used for this game.

[TGM05] = View

Define the camera and view perspective.

[TGM06] = Game Flow Chart

Create a simple flow chart to visualize the course of each game.

[TGM07] = Combat

Define how the different interactions between players looks like and how they connect to each other.

[TGM08] = Story

Define the outlines of a story to explain the gameplay.

[TUI] = UI Design

The game’s user interface needs an initial visual design before being implemented into the game.

[TUIMM] = Main Menu

A simple main menu needs to be design that fits the needs of this type of game. Look for comparable games as inspiration. Not all parts of the main menu will be implemented in the prototype.

The output is an image file showing the concept for the main menu.

[TUIL] = Lobby

The game finding process needs an appropriate user interface to guide the player through it. It should be very basic, intuitive and fit the game‘s theme.

The output is an image file showing the concept for the game lobby.

[TUIHUD] = Ingame HUD

All values of the current game and the player character need to be displayed in-game. Some of them might not be visible at all times though. The user interface should let as much space for the actual game as possible, be intuitive and fit the game‘s theme.

The output is an image file showing the concept for the in-game heads-up display.

[TUIO] = Options

The variable options defined in the GDD must be adjustable through a menu visualized in this task.

Output is an image file showing the concept of said option menu.

[TIMPL] = Implementation

Build a prototype of the game, using the output of the subtasks of [TG], [TUI] and [TA].

[TIMPLG] = Game Core

Implement the main features as described in the concept document.

[TIMPLG01] = Grid Structure

There is a square-grid-style playfield for the players to operate on. Dummy characters can be placed on it and could technically move around, taking into account the game’s rules about movement, defined in the GDD.

[TIMPLG02] = Turn Structure

Requires [TGM01].

There are two players, either on one machine or over network. They take turns to perform their actions. Those actions are not to be implemented at this point. Players cannot win yet.

[TIMPLG03] = Basic Actions

Combines the two previous subtasks to enable player movement. This is an ability every player has and it works exactly the same for each of them.

[TIMPLG04] = Character Template

Implement the character template from [TGC01]. Players would technically use abilities now.

[TIMPLG05] = Ingame Hud

Implement the HUD, defined in [TUIHUD].

[TIMPLG06] = Game Termination

Implement the game termination logic and the logic to determine a winner.

[TIMPLG07] = Terrain Shift

Requires [TGM02].

Implement the terrain shifting described in the GDD.

[TIMPLM] = Additional Mechanics

Implemented all additional features developed until this point. There are no definite subtasks for this task yet as not all additional features have been designed.

[TIMPLC] = Characters

Implement the developed Character classes from [TGC]‘s subtasks.

[TIMPLC01] = Character Class A

Implement the first character class, so players could play a simple one versus one game. Players cannot win yet.

[TIMPLA] = Assets

Implement all assets created until this point.

[TC] = Controlling

Contains all tasks for project management.

[TCP] = Project Plan

Create all necessary documents to give a good overview of the project’s workload and schedule.

[TCP01] = Work Breakdown Structure and Tasks

Create a WBS and a related description document.

[TCP02] = Task Scheduling Table

Assign an estimated amount of time to each task and track the actual time needed to finish it. Add notes for optional or critical tasks. Each task is part of a specific milestone. The milestones should be briefly described at the start of the table.

[TCP03] = Task Allocation Table

Create a table which displays who does what.

[TA] = Assets

This section contains all tasks for assets required by this game.

[TAMO] = Character Models

Each character needs a model to represent it in-game.

[TAMO01] = Character Class A

Requires [TGC02].

Create a model for the character class developed in the associated design step. This includes animations.

Output is an .fbx file of the character, ready to be imported in unity.

# Task Scheduling Table

**First Milestone:**

A flat field (no additional mechanics) with two players, able to move via A\*, synchronized over UNET.

**Second Milestone:**

Implement the height changes in the field and make sure it works with A\*. Implement Jump and height restrictions. Make sure the terrains does not trap characters.

**Third Milestone:**

Implement first heroes (with abilities). Get the point system implemented. Put in the Lobby. Work on everything else that’s not working yet.

**Fourth Milestone:**

Working Prototype. Last-Build. Turn-In.

|  |  |  |  |
| --- | --- | --- | --- |
| **Task ID** | **Estimated Time (in h)** | **Milestone** | **Notes** |
| TGC01 | 2 | First |  |
| TGC02 | 4 | Second |  |
| TGC03 | 4 | Third | Implementation optional |
| TGM01 | 2 | First |  |
| TGM02 | 2 | First |  |
| TGM03 | 2 | Second |  |
| TGM04 | 1 | Second |  |
| TGM05 | 0,5 | First |  |
| TGM06 | 2 | First |  |
| TGM07 | 2 | First |  |
| TGM08 | 0,5 | Second |  |
| TUIMM | 0,5 | Second |  |
| TUIL | 1,5 | Third | Includes possible submenus, overlays and popups |
| TUIHUD | 1,5 | Second |  |
| TUIO | 1,5 | Fourth | Optional |
| TIMPLG01 | 8 | First |  |
| TIMPLG02 | 6 | Second |  |
| TIMPLG03 | 2 | Second |  |
| TIMPLG04 | 1 | Third |  |
| TIMPLG05 | 3 | Second | Might be pushed to third |
| TIMPLG06 | 1,5 | Third |  |
| TIMPLG07 | 6 | Second |  |
| TIMPLM | 12 | Fourth | Buffer time for potentional additional features |
| TIMPLC01 | 6 | Third | Second class is not going to be implemented |
| TIMPLA | 0,5 | Fourth |  |
| TCP01 | 3 | First |  |
| TCP02 | 1,5 | First |  |
| TCP03 | 0,5 | First |  |
| TAMO01 | 4 | Third |  |

# Task Allocation Table

|  |  |
| --- | --- |
| **Task Identifier** | **Group Member** |
| TGC | Konstantin Schaper |
| TGM | Alexander Hoffmeister |
| TUI | Konstantin Schaper |
| TIMPL | David Kuehlmann, Alexander Hoffmeister |
| TC | Konstantin Schaper |
| TA | Alexander Hoffmeister |

****

**Asset list**

In the Max folder there the models that we made.

Robot.max

Turret.max