Ascent Game Design Document

Prototype 2 – Celeste Clone

University of the Witswatersrand, Johannesburg

Game Design IIIB (WSOA3003B)

## Prototype 2 – Team 6

Dominique Bothma (1589523) - Artist

Keaton De Jager (1636214) - Programmer

Stash Gomes (1532956) - Designer

Devan Gray (1615858) - Programmer

Matthew Morris (709058) – Project Manager

A platformer about climbing out of the underworld as you learn to persevere, learn and overcome.

# Contents

1. Game Overview
   1. Game Concept
   2. Key Features
   3. Genre
   4. Target Audience
   5. Look and Feel
2. Mechanics, Dynamics and Aesthetics
   1. Mechanics
      1. Basic Movement
      2. Jumping
      3. Wall Grabbing
      4. Wall Climbing
      5. Wall Jumps
      6. Dashing
      7. Death
      8. Jump Pads
      9. Spikes
      10. Collapsing Platforms
   2. Dynamics
      1. Learning through failure
      2. Experimentation as Tutorialisation
   3. Aesthetics
      1. Movement as Fun
      2. Reliability of input
3. Level Design
   1. Level 1
   2. Level 2
   3. Level 3
4. Visual art, narrative and Sound
   1. Art
      1. The Player
      2. Level Art
   2. Narrative
   3. Sound
      1. Music
      2. Sound Effects
5. Technical
   1. Development Hardware and Software
      1. Hardware
      2. Software
   2. Class Diagrams
6. Known Issues and Technical Constraints

**Game Design Document - Celeste Clone**

**Game Overview**

The role of this project is to create a clone of another game. For this prototype we have chosen Celeste. Our rendition of celeste will feature identical mechanics and a new aesthetic and art style.

Celeste also provides an interesting level design challenge. Celeste’s mechanics are difficult to get an initial grasp of. As such tutorialisation through level design is important.

We have chosen Celeste due to the depth of its limited, but complex mechanics. This is both to learn better coding practices, teamwork between multiple programmers and to provide a solid case study to draw from for our individual game analysis reports.

**Mechanics Breakdown**

Controls

|  |  |  |
| --- | --- | --- |
| **Function** | **Keyboard** | **Controller** |
| Moving (also directs the dash) | WSAD | D-Pad or Left Analogue Stick |
| Jump | C | A or Y |
| Dash | X | X or B |
| Grab | Z or V or Left Shift | LB, RB, LT or RT |

**Basic Movement**

1.       Basic Left and right movement – basic moving left and right. This is precise. It sets velocity. There is no friction, no acceleration or inertia. This is straight velocity. Needs to be a check for if there is a wall currently in front of the player (while grounded) to prevent them from walking into the wall. If the player is airborne and walking into the wall they enter the sliding state.

2.       Jumping – Should include the “hold to jump higher”. If the player presses the jump key but can’t currently jump (due to being airborne) then the jump is queued for a very short duration. If the player then meets the criteria to jump, such as sliding down a wall or being grounded) then they jump. This way the player can press the jump key just before landing but still be able to jump. There also needs to be some coyote time, where the player can jump shortly after walking of a ledge.

3.       Grounded and Wall Collision - Detecting whether the player is grounded. If the player is airborne **but** moving towards a wall, they slide down the wall, falling at a reduced speed. Letting go of the movement will return the player to their falling state. Holding the grab key will place the player in the grab state.

4.       Wall Climbing – If the player is colliding with a wall, holding grab will stop their fall. They can hold this grab indefinitely. Pressing up or down allows the player to climb.

5.       Wall Jump - There are 4 ways this works.

1. If the player is falling next to a wall, the jump will push them up and away from the wall a short distance. Holding the jump key longer still increases jump height, but not the distance the player is pushed from the wall.
2. If the player is sliding down the wall, jumping will push them up and away from the wall. The push away will be a greater distance. Holding the jump key longer still increases jump height, but not the distance the player is pushed from the wall.
3. If the player has grabbed onto the wall, then jumping will push them up the wall a small distance. Holding the jump key increases the jump height. I think this works by temporarily disabling the grab ability.
4. If the player has grabbed onto the wall and facing away from the wall (diagonally included) then they jump as if it’s case 2.

6.       Dash – Works in 2 ways

1. While airborne - the player can press dash to shoot in one of 8 direction (up, down, left, right, diagonals). This is determined by which directions on the d-pad are currently being pressed. If there is no directional input, then the dash defaults to the facing direction. If the player is holding grab while dashing and then collides with a wall they immediately grab. Onto that wall.
2. If the player is grabbing onto the wall - the player still dashes as normal. However, grabbing is disabled and reenabled at the end of the dash. This allows players to dash off walls while grabbing, or dashing further up a wall.

Once we have this all working, we can start adding limitations to these mechanics.

**Limitations**

1.       The player can only dash once while in the air. This is refreshed if the player lands on the ground (including disintegrating platforms, one way platforms, jump pads), or if they hit a camera transition, or if they collect a jump refresh pickup.

2.       The player can grab indefinitely but moving and jumping is limited. While grabbing, the player will have a grip points system that refreshes whenever their dash resets. Moving up and down costs points (based on distance moved, so that moving into a ceiling doesn’t deplete it) and jumping costs points. Holding also costs these points, however the number is only.

**Environmental Obstacles**

1. One-Way Platforms - The player may jump through this platform from the underside. However the platform acts as a regular platform from the top.
2. Spikes - The player dies when they land on the spikey side of this platform, however they can jump through the underside of the spikes.
3. A moving Platform - When the player lands on the top of a moving platform, or grabs onto the side of it, it moves. If the player is on the top of the platform, or grabbing onto the side the player’s velocity is increased/decreased by the platform to keep the player in the same position relative to the platform. While the player is moving on a platform, if they are squished into a wall, they are killed.

**Pickups and Rewards**

1. Collectables - Like the strawberries in Celeste, Touching a strawberry makes it follow you, being collected only when the player touches the ground. We don’t really need to store whether or not this strawberry has been collected previously. It’s just a prototype.
2. Jump/Stamina refresh - Colliding with this object will refresh the player’s stamina and allow them to dash again. These respawn shortly after they have been collected,

**Misc Stuff That is still really important**

1. Camera Bounds - The camera needs to be able to snap to the bounds of the level, but also be able to follow the player if they are in a larger area due to verticality or horizontal distance. We can keep the zoom level the same throughout the levels. Snapping to rooms will be important though.
2. Death - When a player dies, they will perform a short death animation and after a brief delay be returned to their checkpoint and the stage will reset. This will include any collapsed platforms or consumed jump crystals.
3. Player Checkpoints - When the player dies, they will be returned to their last checkpoint. Checkpoints are invisible and can be assigned when a player moves into a new room, or reaches a key area in a live.
4. Level and area management - When moving from one level box to another the player’s stamina and dash need to be refreshed. THey also need to be assigned a checkpoint. However, this needs to be specific to the entrance they used.

**Assets List**

**Character Animations**

Note: There will be two sprite sheets for the player. This will be with powered and unpowered prayer beads to signify whether or not the player can currently dash.

1. Walk Cycle
2. Idle Loop
3. Wall Grabbing Idle
4. Wall climbing
5. Wall Sliding
6. Jumping (falling and rising)
7. Dashing (you can make up, down, forward, downward-forward and upward-forward variants if you want to be fancy but run this by Keaton first to see if that's possible).
8. Death

**Environment Art**

1. At least on generic environment tileset
2. Maybe darker background variant of this generic set for some background decoration (this could also just be done in unity though by tinting the canvas slightly dark blue)
3. One way platforms
4. Spikes
5. Collectable (strawberry equiv)
6. Jump Refresh

**Particle Effects**

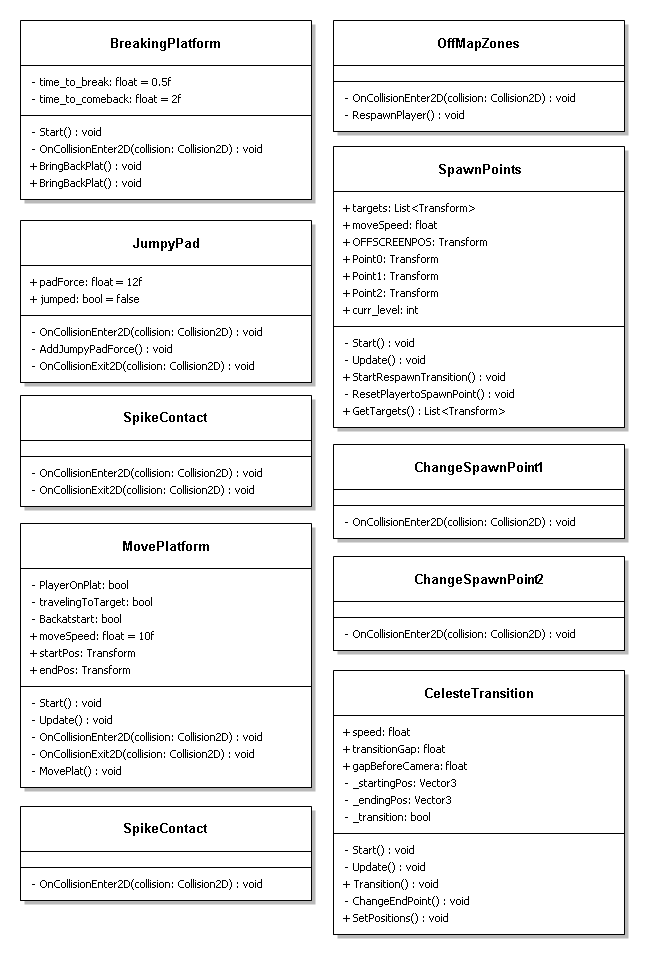
1. Dash Effects
2. Landing effect
3. Jumping effect
4. Sliding Effect
5. Collectable collect effect
6. Dash refresh effect
7. Death effect
8. Respawn Effect

**Sounds**

1. Walking sound loop (probably optional, they’re hard to find… and can be really annoying if you find a bad one)
2. Jump sound
3. Land Sound
4. Dash Sound
5. Collect strawberry sound
6. Collect dash refresh
7. Death Sound
8. Revive Sound

**Class Diagrams**

**Environment Scripts**



**Player Movement Scripts**