Escape Time!

Version 2.0.0

Justin Kephart
Game Design Technologies
Spring 2018
Tim Samoff

DESIGN BRIEF

Introduction

You are a treasure hunter who has just made the discovery of the century. . . a pocket watch that can slow down time! Time to get out of here. Wait, what's that grinding noise?

Design Statement

The name of the game is a double entendre, as it is not only time to escape, but the player is effectively able to escape the constraints of time itself.

This setting was much easier to design puzzles for, using the slow time mechanic, which is why I completely remade the game. I had a lot more fun designing this one.

I figured since I was compacting the mechanics into tileable blueprints, I might as well make the level randomly generated.

Audience and Context

Fans of runner games will enjoy this. It takes some skill to make it out.

Core Gameplay

Pick up the watch with E to gain the ability to slow time! Use RMB to slow time. This starts the level immediately however, and you need to start moving.

The starting room and ending room is predetermined, but most of the tiles are generated using an algorithm. The amount of tiles generated depends on the difficulty chosen at the menu.

Escape before you get killed! Dodge traps and jump gaps. The level generation has been designed to not give you room to stop. Boulders come immediately after any series of pits or walls that could block a previous boulder. Generation is also prevented from placing two peaceful rooms in a row.

+

LOOK AND FEEL

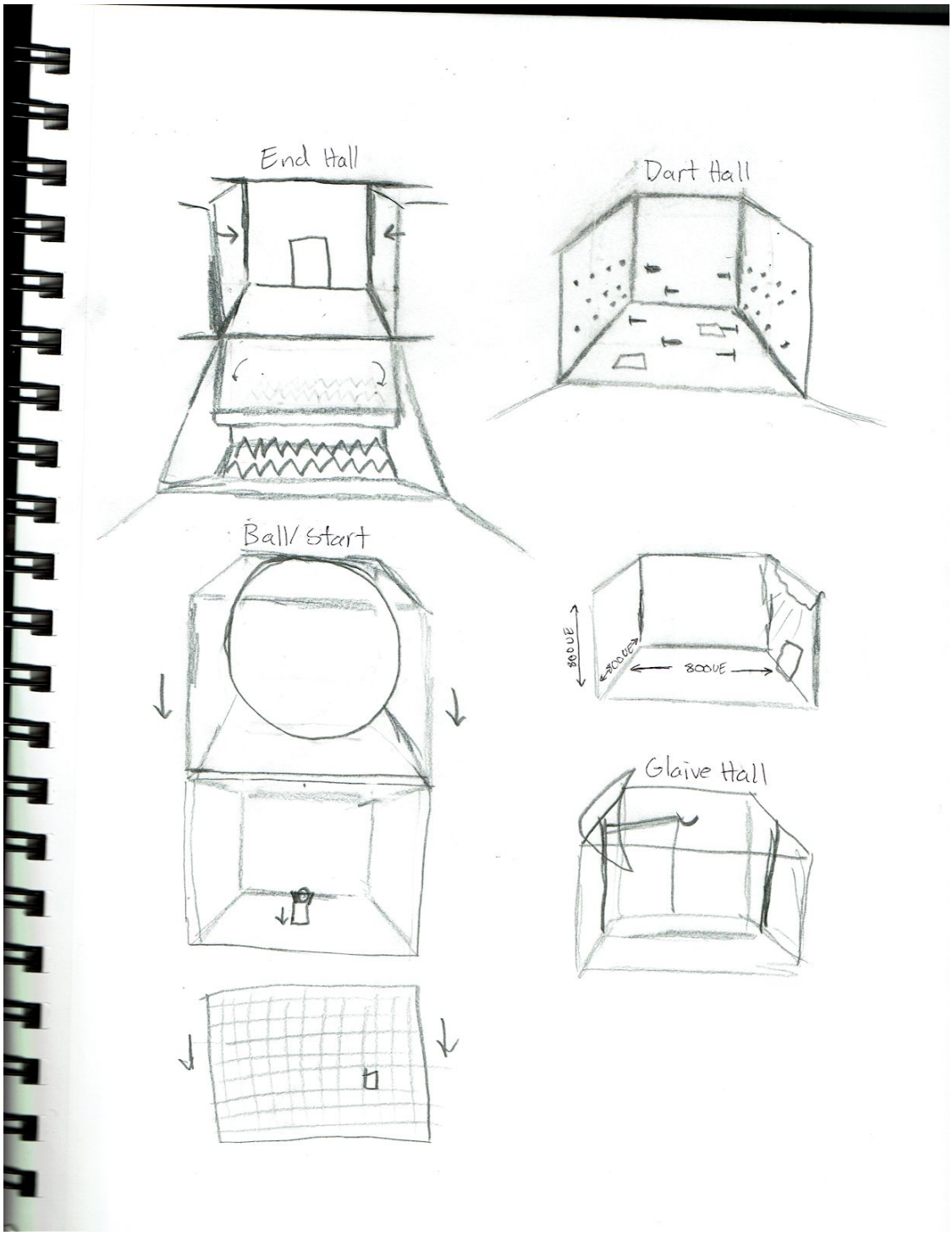
This game's design is obviously designed after Indiana Jones. The golden artifact and weight triggered trap is emulated in the game.

The boulder and some of the traps are ripped straight from the movies. I imagine Indy would have had an easier time with an artifact that could slow down time!

Quicksilver from one of the the latest X-Men movies was also the main influence behind the time slowing mechanic. I originally wanted to make an FPS with the mechanic.

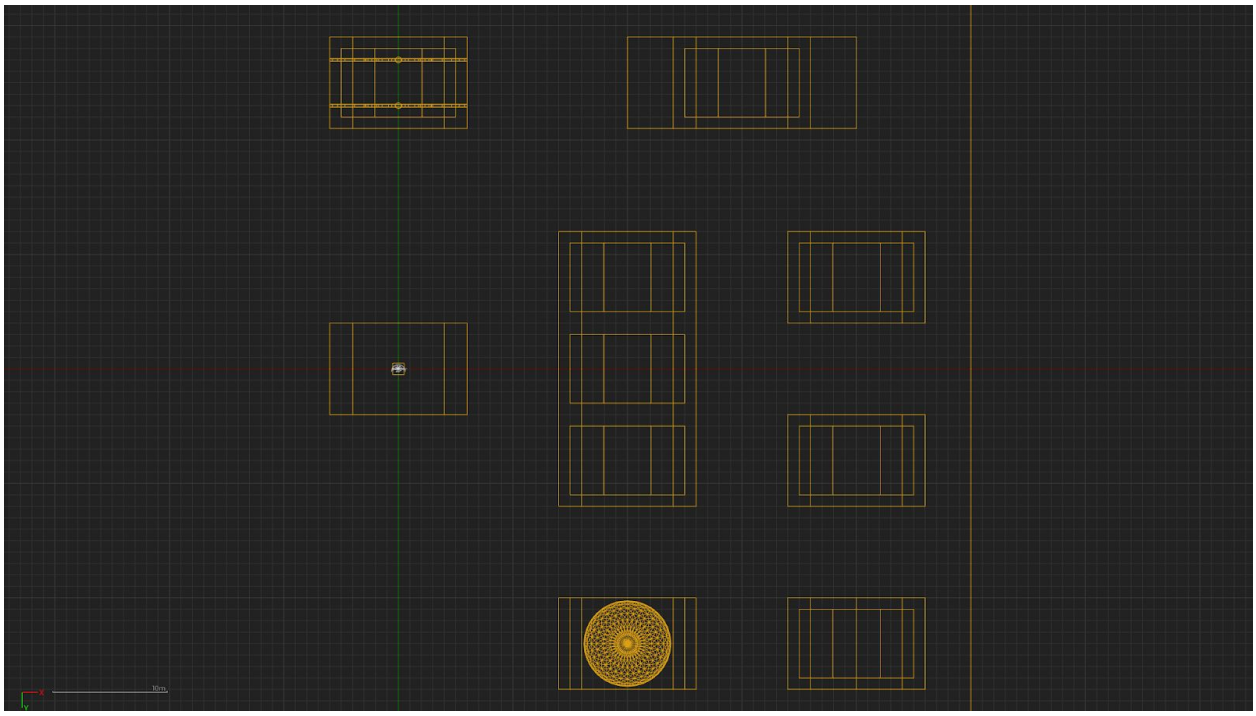
MAPS AND GAME FLOWS

Initial Hand-drawn map



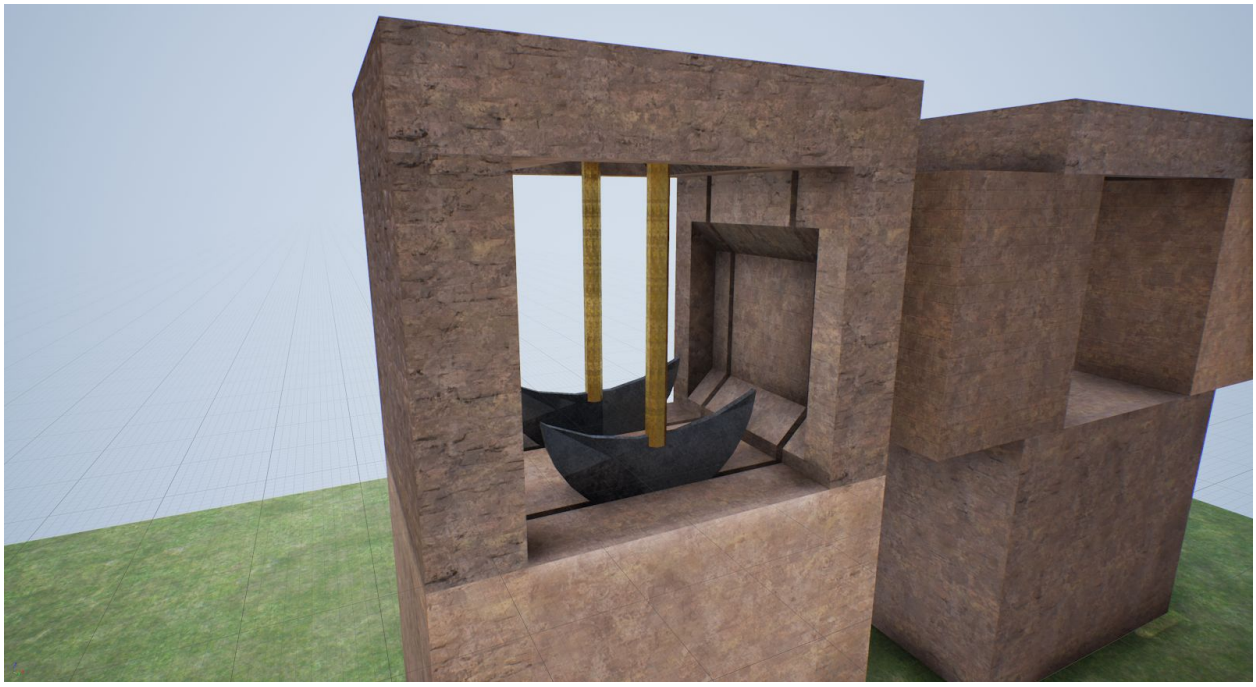
Basic ideas for traps. Need to experiment with the actual dimensions in engine.

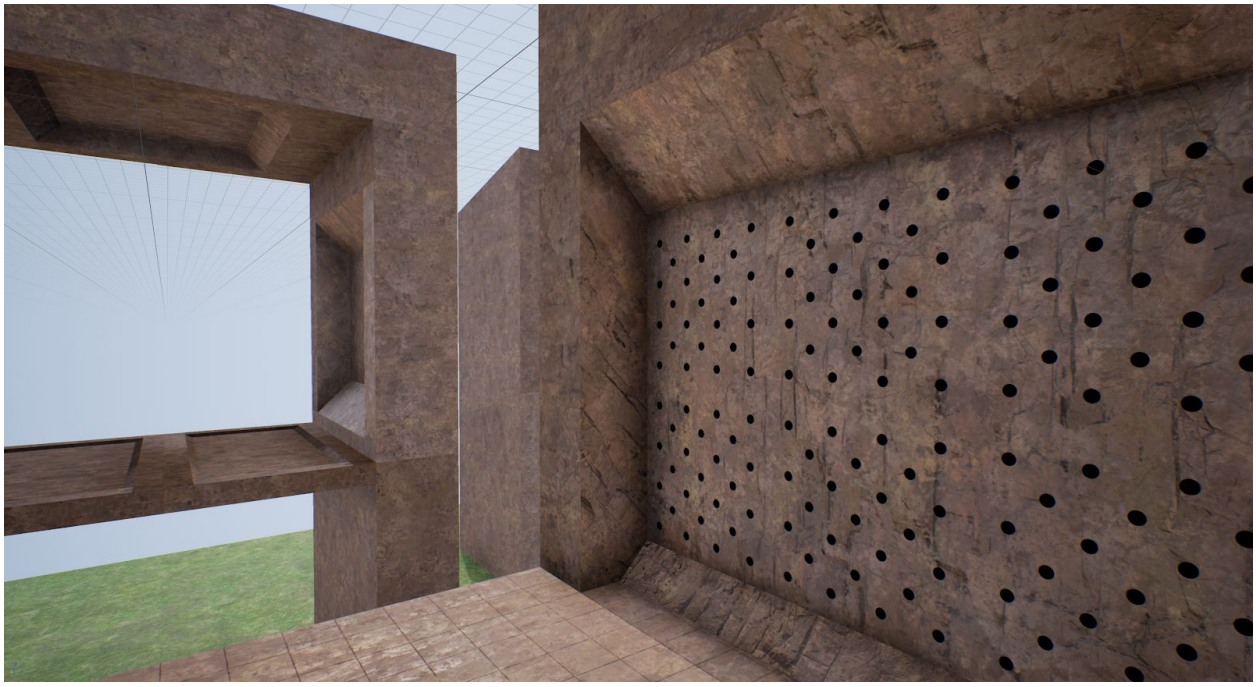
Creating the Assets

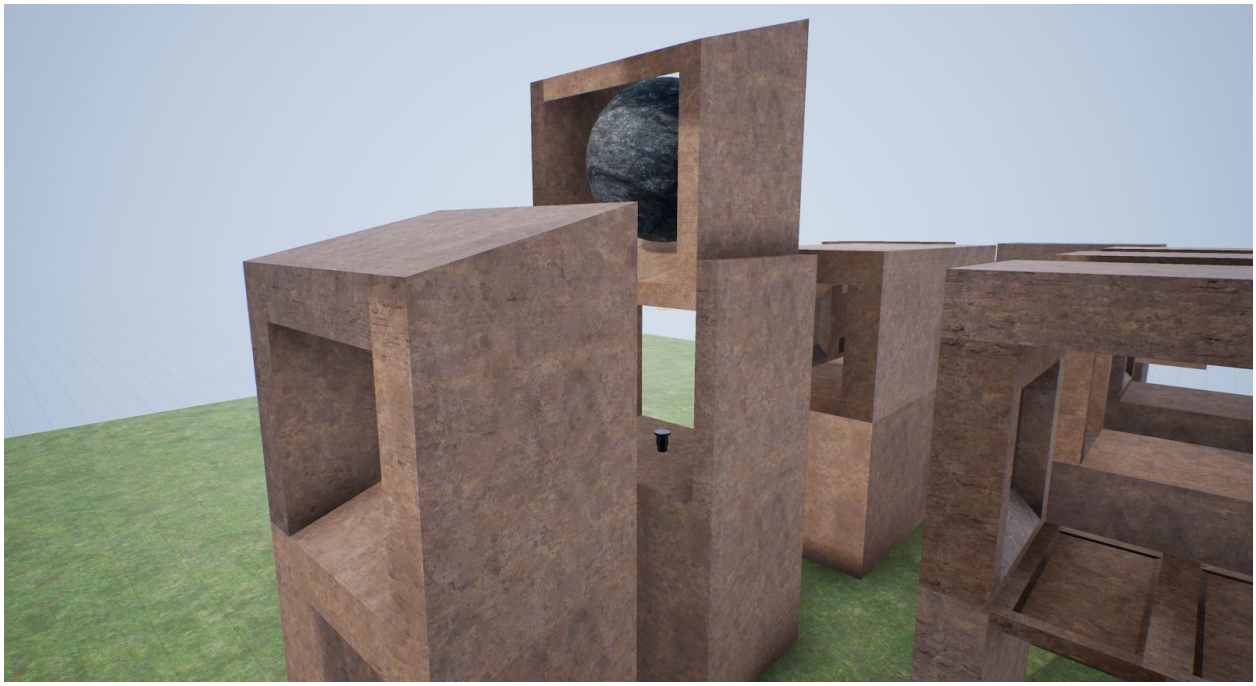


I made a map to construct the modules and objects out of geometry. Then I duplicated it before converting to static meshes (since it's only one way)

Finished Meshes





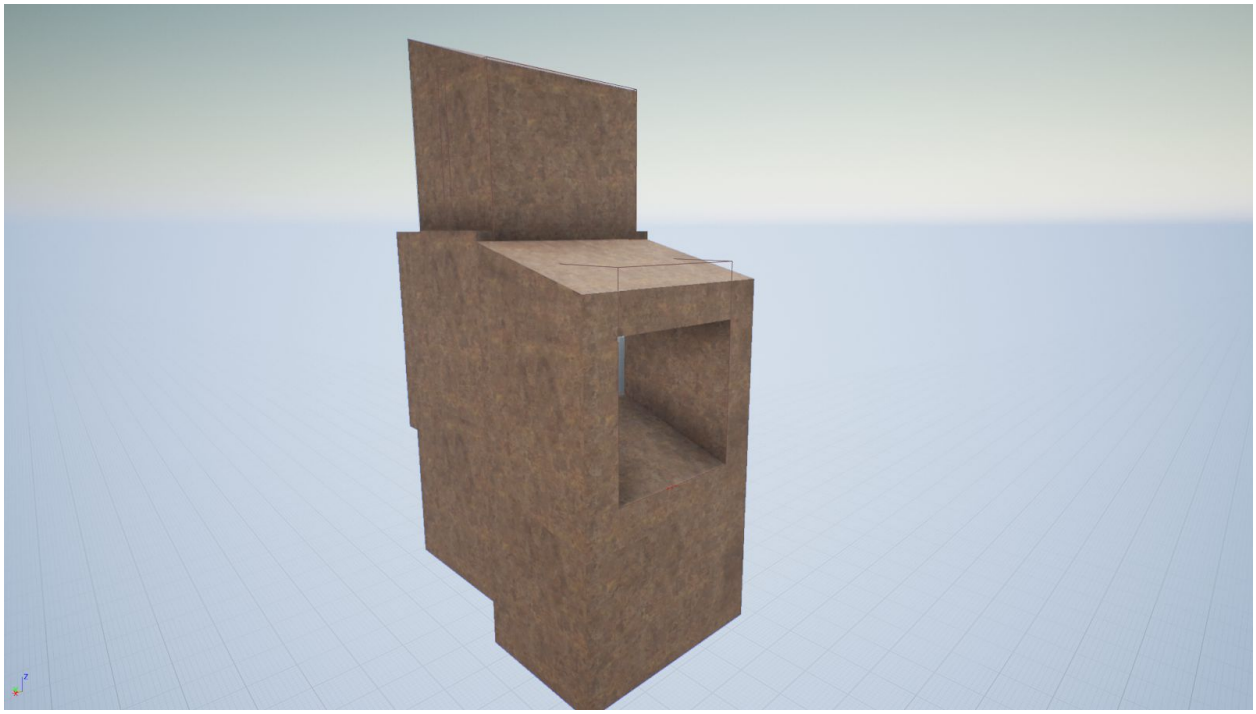


When I was done with my meshes, I assembled them into blueprints and made their individual logic.

Main Menu Map

I figured I'd use the end piece for a main menu since you have to enter some way.

Game Map



This is what the main map looks like before the level starts. As soon as it starts, the level is constructed with the algorithm. Inside is the pocket watch.

CREDITS

- Stone Moving Sound: “Stone grind” sound from Dissonant Introspections by Katarrhaktes is part of the Xperimental Sound Lab, a free sound collection curated by Thalamus Lab.
- Boulder Collision Sound: “stone dragging on stone” by thanvannispen
- Crush Sound: Snap-a-neck03 by CGEffex
- Menu Sound: Boom Band by bareform
- Firebrand Kevin MacLeod (incompetech.com)
  Licensed under Creative Commons: By Attribution 3.0 License
  http://creativecommons.org/licenses/by/3.0/
- Prelude and Action Kevin MacLeod (incompetech.com)
  Licensed under Creative Commons: By Attribution 3.0 License
  http://creativecommons.org/licenses/by/3.0/

Final Gameplay Scenes:

Post-Mortem:

So first, I should talk about how I essentially made this game twice. I started off with and idea for a first person game using slow motion mechanics to gain an advantage. I started designing a sci-fi FPS game to use the mechanic, but I found out I was not allowed to add AI or hostile weaponry. I tried to make puzzles for this space ship sci-fi adventure, but for life of me I could not think of any. Spikes and traps really have no place on a spaceship. I was having a horrible time, barely making any progress inside or outside of class, so I decided to just start over and make this.

Now I was down a few weeks when I started this, so I had to find a way to make a difference. I believed object-oriented programming was the solution, or in UE4 talk, blueprint-oriented game design. Then I thought, as long as I'm making my level pieces modular, why not randomly generate the map to increase replayability?

As you can probably guess, this game required a lot of knowledge that wasn't taught in class. It was hard at first, but once I got used to the blueprint system, it wasn't so bad. Youtube videos and pictures of "code" helped a lot. Alex Hoffman was also a lot of help. He taught me a lot of techniques to use in UE4. The most important thing was probably timelines. They are very versatile and can be set to ignore the slow mo effect. Designing the code was the most fun part of making this game.

My main problem was assets. I am not an artist and I don't know how to 3D model or rig, so I took a lot of assets from the Starter Pack, Animation Pack, Soul Cave pack and modified what I could. It was pretty much trial and error for that since I didn't know much of the terminology either. I only grabbed two models off of the internet: the pocket watch and the helicopter, and I had to texture, assemble, and animate the pocket watch.

All in all I learned a LOT about UE4, which is the main reason I took this course, so that's great.

Conclusion/Future:

This was a really fun project to work on once I started over. Since I made the game modular I can freely add new modules and they will work with the level generation. Making new modules is hard work though. I might come back and work on this. I think this would be interesting in VR.