# Planning

**09 January =**

* **TEST WITH ZHENGA**
* **Items =>**
  + **Add new items I invented** (sprite + functionality + tutorial)
  + Add the “beam” option to other items that might need it? (And “unit beam” or “max”.)
* **Soundtrack**
* **Finalize marketing items + put on Itch.io (and other pages)**
* **See if we can make things more robust** (go through TO DO/DEBUGGING)

**10 January =**

* Record BG music trailer (while we also finish the other album)
* Write down all future to-dos and anything else I might need later.

Odd bugs:

* Teleporting => the “old\_room” isn’t properly reset to null after teleporting, sometimes?
* Terrain erasing/overpainting isn’t *flawless*, probably because that system changed (for saving/determining which room a cell has). Is it a problem?

**?? =>** playtest, cut to trailer/GIFs/screenshots, completely finish

# To Do

## Special Tiles

**TO DO:** Write code in player manager to *always know the order of a player’s bodies*.

**TO DO:** Find a good, nice implementation for the freeze beam idea

A tile that freezes bodies nearby *if they aren’t your worst body*.

* + This might seem an obstacle, but it’s actually a helpful thing.
  + It locks a body, so you can safely move your others without worrying about it.

?? Freeze Beam: everything inside just slows down a lot (area physics override gravity and damping?)

**OR JUST COMBINE THIS WITH THE “freeze everything but your worst?”**

If it slows you down considerably, you have time to bring your other bodies forward

**IDEA:** A special tile that has a *spinning* component (which basically acts as a racket to slam players away.)

### Misc

* Create tutorials for all new special tiles
* Actually divide coins among bodies when they are spiked (or do I already do that?)
* Things that show a beam: color (and alpha/modulate) the beam accordingly!
* Other types of platforms? (One thinner, one smaller, one with a hole.) (For variation *and* because the current size isn’t great in all situations.)

## Re-Test

* All the new special tiles
* (+ Changed tiles, such as shield and spiderman that got a *radius*.)
* Glueing
* Probability-based drawing

# Future To Do

**Also teach the other controls on gamepad (LT/LB and RT/RB)?**

**Glueing still seems to reset your size to the wrong value?**

**Maybe the “modify jump normal to match surroundings/wall jump)** **is a bit too strong/prevalent?**

**Simple Generation mode (used in tutorial course):** Add more control over first few rooms. (Keep flat, ensure a jump is needed, etcetera.)

**Better labels** => somehow, ensure they are placed *within the room* and without *overlapping important stuff*.

**Magnet particle/visual =>** create white circle moving inward?

**Shape change particle/visual?**

**PRETTIER GENERATION:** Add random elements, like grass, flowers, wooden fences, in foreground and background.

~~When you are a non-standard shape, make the timers (for rounding/malforming) a bit~~ *~~longer~~*~~, so you stay in that form for a while.~~ => probably a bad idea, as it’s hard enough as it is, and it would just frustrate players with no way out

**BUG/IMPROVEMENT:** Sometimes there are still really tiny gaps to move through, otherwise you just can’t progress => It’d be best to eliminate them *for sure*

More items specifically useful in *solo mode*?

**BUG:** That bug with the entities array containing a “previously freed instance”. Theories:

* Happens after “stood still too long” teleport?
* Happens when sliced at some odd moment?

## Map Improvements

**FILL ROOM Algorithm:** Add a variation where we’re allowed to place tiles *against the walls*, but *not in the center*. (By default, we only place away from walls, in the center.)

**Problem?** Should find a way to ensure that connections to other rooms stay open.

(Before placing, check if this tile connects to a different room. If so, don’t allow it.)

(When placing a new room, also recheck the tiles on the previous room?)

**3-way-open-tiles removal:** Create option to *replace* them with something instead? (Now there’s a chance of breaking walls and ruining the generation.)

**Tutorial generation:**

* Randomly remove slopes at the *top* (or bottom, for that matter) to allow for bigger and cleaner openings => especially in **simple generation**/tutorial

Find **“annoying patterns”** and fix them in varied ways.

**Special Tiles/Items:**

Think of more fun stuff that plays with **blocking players doing really well + physics variation**

* Create platforming sections (with ramps, moving platforms, etcetera)
* Create other inner gates. (Like the laser/cannon.)
* **IDEA:** Some beam/magnet that attracts *across the whole column*. (Shoots across the room like the laser.)
* **IDEA (related):** A fan that blows ( = repels) *across the whole column*.

**Lock:** **Shop**. (If I figure out a good way to do it.)

**Add *special tiles* that do something with coins!** (And once we’ve done that, add an area to bodies, which lights up the coin interface when such a “coin\_related” tile is in range.)

**INPUT:** Add different control scheme for controllers: joystick to roll left/right, any button to jump/float.

* (Make this default? Or can players configure it themselves?)

**GENERATION**

* Play with generation parameters => I feel big rooms should be *slightly* less filled (or have more varied filling), maps should *flow* a bit more (with slopes, rooms that are not *too* different in size/displacement)
  + “Preferred” displacement would be something that does NOT create a bump in the line. So either it stays flat at the ceiling, or it stays flat/falls down on the ground.

Don’t put a **paint lock** on a *huge ass room*.

# Trailer

Parody of “Rolling in the Deep” (by Adele).

## Footage

**Record footage from playtest.** Accompany background with footage that complements what’s being sung.

* Under 90 seconds
* Start with one attention-grabbing, supercool intro. (During intro of song, with just the drum. Or maybe even before the song starts.)
  + Start with your best joke, end with your strongest material.
* Then go slower, and build it up again, until the final climax.
* Use (animated) text and titles that fit the game.
* **Make sure they can easily be cut into cool GIFs => spread those as much as possible.**
* **Record footage *without* BG, so sound FX come in the trailer.**
* Make first 2 lines of YouTube description the best => those will appear in search results

Game trailer templates

* **Tell, Show, Repeat:** use title cards/narration to tell a thing … then show the thing. Repeat. It’s best to call out what you *do* in the game or what your *goals* are. (Don’t call out *raw features*.)
* **Music Video Montage:** Take fun gameplay and cut it to the beat of the music. Best used for games which are *very simple to understand from just watching*. Can also sequence gameplay so the *simplest shots* are at the beginning and you slowly build.
* **Chronological Order:** record footage of gameplay, then simply keep them in that order but cut them in some exciting way. Many games are structured like that: introduce ideas, then explore some twists on the idea, then test player’s ability to understand them. So keep that sequence to entice the player.
* **Just Explain The Game:** some games are really hard to understand at a glance. So just use text/title cards/narration to explain the game in a linear fashion. (Then use good music and editing to make that exciting.) Mostly works for games with visuals that are hard to parse/understand in a fast-cut trailer.
* **In A World …:** story trailer. Say in this order: 1) this is the world/premise; 2) this is the *person* in that world; 3) this is the *problem* that person faces; 4) this is how they *confront* the problem; 5) these are the *obstacles*

Make sure footage is **clear:** no HUD or UI, or just compose shots to have a clear focal point (or limited number of things on-screen).

Retention of knowledge/information is far more important than quantity of content.

## Lyrics

### Verse

There’s a strange folk

They are called the Sheepe

Weird shapes and biting wolves

They roll and jump and leap

There’s a game

For 1-6 players

Roll to the finish first

Before your sheep gets hurt

### Pre-Chorus

The world is always

Different every game

To keep you thinking:

Boy, when have we seen it all?

You only need to know

Two buttons total

No time for blinking

### Chorus

You could just win it aaaaaaaallll

Rolling in the Sheeeeeepe

Enjoy the game with aaaaalll of your heart

Now go play, go play it, it is cheap

**Not mentioned in lyrics =** local multiplayer

## Discarded Lyrics

There’s a game

Two buttons to learn

Roll left and roll right

But don’t take the wrong turn

# Done

## Annoyances

**ANNOYANCE:** When you jump with your head against the ceiling, your *rotating* movement actually pushes you in the wrong direction. Which is just … annoying? (Yes, you can learn it, and use it for stuff, but … not great.)

* Solution #0: Make ceilings frictionless => can’t do it, as they’re part of the tilemap, which has *one* physics material.
* Solution #1: Always cling to ceilings => possible (check if cling vector is opposite to gravity vector)
* Solution #2: Make jumping less powerful
* Solution #3: *Hold* both buttons to *float* or *steady yourself*. (So when you hold both, your Y-velocity becomes 0. But your X-velocity continues.)

## Basic Bodies

**Step 1:** Generate a random polygon

* <https://stackoverflow.com/questions/8997099/algorithm-to-generate-random-2d-polygon> => basically, create a circle, but allow each point to vary in radius/angle
* <https://stackoverflow.com/questions/59287928/algorithm-to-create-a-polygon-from-points> => draw a point cloud first, order by angle, then draw through it

**Step 2:** Calculate its centroid. Place a smiley face there. Then center the polygon around it.

**Step 3:** Turn it into a physics body + draw it each frame.

**Step 4:** When given input, roll in a certain direction. (Check if this actually works for movement.)

## Body slicing

**Step 1:** Write the slicing algorithm I scribbled on paper.

* <https://stackoverflow.com/questions/563198/how-do-you-detect-where-two-line-segments-intersect> => detect intersection point of two lines
* The rest of the algorithm is just:
  + Loop through shape.
  + Detect first intersection point. Add it to the shape. (Between the start/end vertices of the edge it intersects.)
  + Continue until second intersection point. Add it to the shape.
  + Now *extract* the part between the two points: shape 2. *Remove* the part you extracted from the original shape: shape 1.
  + Now recreate the *bodies* + *draw/move scripts* for each.

**Step 2:** Allow testing by drawing with the mouse. (Or clicking twice. Or pressing a key and testing a predefined line.)

**Step 3:** If successful, allow applying dynamically.

# Discarded

The old idea with “placing precreated rooms”

## Rooms & Routes

**Issue 1:** How do we allow *rotating* rooms?

* Translate everything to anchor center
* Rotate the thing
* Translate everything back => DOESN’T WORK, because the “position” property is still local, so translating back would just *follow the new orientation*
* Now recalculate opening values

**Issue 2:** What if a single side has *multiple* openings?

* We should be able to match any of them
* But *not* necessarily close the others when filling gaps

**Issue 3:** Now we have ugly *double walls* between rooms.