# Planning

**03 January =** settings/performance optimizations

**04 January** = feedback, particles, sound effects, animations, etc. + **SOLO MODE**

**05 January** = fix crucial bugs, answer essential questions, really finish and make playable

**06 January** = record BG music + singing for trailer => make sure all pages, content, etc. are ready

**07 January =** extra day for extra content, more route variation, fixes, etcetera

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

## Feedback/Particles/Sound FX

**Movement:**

* Hit something (thud)
* Rolling over stuff => maybe only when ice powerup is active (or something)
* Jump
* Float

**Game loop:**

* Teleport
* Finish
* Lock *unlocked*

**Status:**

* General => when shape is changed
* General => teleported
  + When this happens for behind-the-scenes gameplay reasons, explain *why* it happens. (“Stood still too long!”) (“Too far from leader!”)
* Do/undo ghost
* Do/undo invincibility
* When body is destroyed, but *not* as part of a slice.

**UI:**

* Button Click
* Feedback when player logs in and stuff in menu

**Locks:**

* General => “lock progress”
* General => “press a button”
* Collect coin
* Painting/Erasing
* Gate => open/close

**Terrains:**

Many terrains change physics/interactions with tiles. So just make something *different* for these ones:

* Ice
* Spiderman
* Glue/Spikes => show something around players
* Magnet => show something around players

**Special tiles:**

* Explode
* Hit by bullet + bullet shot
* Hit by laser
* Trampoline jump
* Speedup/Slowdown (powerup/powerdown sound)
* Freeze
* Time bonus/penalty
* Fast forward/backward => or is this covered by the teleport effects?

# To Do

## Performance

Now I access modules at runtime, in each script separately, with get\_node.

Instead, use get\_node() in the *main script*. Then use body.<module name> to access it.

Would prevent *many* expensive tree calls.

## Solo Mode

**How does this work?**

* The map is slowly dissolving behind you
* If it reaches you, you lose
* If it swallows one of your parts, you get a penalty.
* Just try to reach the finish.

## Coins

Bigger question: what exactly are coins good for? (It feels a bit tacked on at this point.)

* Coins protect you from the wolf. (They just take a coin, instead of biting you.)
* Some locks require them *or* make your life easier with coins.
* Some terrains use them.
* Some elements should also require payment? (Make sure it doesn’t become overly complicated!)

**TO DO** => Mark certain items as “coin” => show coin interface when *near* them

* Use the physics for this? Or just do a loop through the 3x3 area around us in the grid?

## Big Question

What’s the point in *slicing bodies*? If only the first one needs to finish?

* Make you smaller => which might or might not be good.
* Individual bodies have fewer coins => which means fewer possibilities.
* **The more bodies of yours that finish, the more *time bonus* you get?**
* Stray bodies might activate something you don’t want. Or be an easy target for a wolf?

How to make your *starting shape* matter at all? Make it more probable that you are *reset* to it along the way? Only make “bad” starting shapes and, when the wolf touches you or something, you revert back to that?

## Bugs

**BUG:** When continuing from backtracked room, lock doesn’t open the right connections (or any connections at all) to previous room

**ESSENTIAL BUG**: When backtracking, it allows overlap with older rooms? Which messes up everything BIG TIME.

**BUG:** Elements on “weird” tiles (such as the one open on 3 sides) are oriented in the wrong way.

**BUG:** When wolf slices you … sometimes only *one* body remains? Instead of the expected two?

**BUG:** The “holes” painter sometimes narrowly overlaps tiles inside. We can’t erase those (as they are solid), locking the game completely! (I already added code to “fix” this … but because of the rounding and stuff, it’s not perfect.)

**GATE BUGS:** “Recalculate\_gates” => we don’t remove the old gates! We just add/set new ones if necessary. Fix that.

Such as the “sacrifice\_coin\_gate”, not sure though.

Also not sure if I added the automatic coin spawning to those gates to ensure you’re not stuck forever.)

**BUG:** Sometimes it counts collecting a coin as collecting *two coins*. (Sometimes even three???)

**BUG (?):** When glueing bodies back together, their scale is all wrong?

**PERFORMANCE:** When the whole map painting texture is active, performance isn’t great. How to improve that?

* Add a “performance mode” in the settings that turns it off.
* Only draw the *currently used area* of the screen?
  + It calculates the rectangle from top-left of the route to bottom-right.
  + It only updates and draws that, though with some margin.
  + Once in a while, it shrinks/grows if necessary.
  + *This might not even be the issue, as the second “mask” texture is huge*.

## 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?)

**Tutorial generation:**

* Add final tutorials + more control over first few rooms
* Randomly remove slopes at the *top* to allow for bigger and cleaner openings

**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.)

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

## Polishing

* 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.
* Whenever I do something to a jump normal, show a tiny line for that. (Similar to debug, but prettier.)
* **Extra rule:** When one of your bodies does something or gets something, this is copied to ALL your bodies.
* “Coin lock” => perhaps too many coins, that appear too quickly
* Add “bouncy” tween to the *shaper* node when jumping or hitting stuff.
* Give an indication when someone is holding *both buttons* => perhaps show a different color or overlay when *floating*. (Add wings at the side?)
* Give feedback
  + Especially when getting a time penalty or getting/paying coins.
  + But also enable option to give *textual feedback* when someone first enters a terrain, which gives a hint *how* this terrain works.
* 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?)
* **Currently, tiles are only placed on the *inner* rectangle of a room.** This is only needed for (button) locks, as otherwise it’s not bad if items are off a bit. So remove the restriction on any other room?
* 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.

# 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.