# To Do (Sheepe)

## Leftovers

**Terrain:** magnet?

**Terrain:** each time you enter, an old body of yours is removed (if it exists). (Alternatively: for every X seconds you stay here. Might be harder to explain and track, though …)

## Teaching

Make the **tutorial** course => plan tutorials once in a while (fixed order and distance between them), only activate a rule (such as the wolf) once the *last* player has passed it. (Mechanics and stuff are just activated always.)

## Locks

* **Shop (with modification?)**

## 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.

## Gameplay Improvements

**GREAT IMPROVEMENT:** When inside a lock, you are also prevented from *moving out of it*. (An invisible gate is present that forbids going back.)

Why? So it’s less annoying. So you have a “safe place” and can take time to get back your older bodies.

Making coins useful:

* If you have coins, the wolf does not *split* you, but rather takes a coin from you. (Both makes the wolf more interesting to play actively, and gives anyone a use for coins.)

Making body splitting useful:

* More “glue\_related” things, most likely items
* Older bodies of yours can still activate stuff, so they can still be useful (or very annoying) if you keep track of this. (For this to work, we need one other rule: *when one of your bodies does something or gets something, this is copied to ALL your bodies*.)

## 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?

## Bugs

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

**BUG:** You’re not teleported forward if you’re standstill inside a *lock*. (I probably only check for terrain, not if the lock itself still exists. Fix that!)

**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.)

**BUG:** I *might* have broken gates. (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.)

General\_parameter is, somehow, not passed to gates (or overridden later).

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

**BUG:** Don’t add inside tiles into a “tutorial” room? Or decrease probability and make the tutorial slightly opaque?

**ISSUE:** It’s annoying if the wolf teleports forward (because of standstill or something) … and immediately slices people. Feels like, when the wolf is teleported, it should be removed using the current system. (So that it takes a few seconds before it switches to someone else.)

**BUG:** It still sometimes adds two locks right after each other. (Probably has to do with something being “lock\_planned” but the lock isn’t there yet.

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

**BUG:** Sometimes item (spikes) not removed after hitting it? (Probably has something to do with being unable to find the item in that location, there was something wrong with that, wasn’t there?) => **might be fixed**

**BUG (?):** When glueing objects back together, it sometimes crashes? Because the room.players\_inside array has a few null entries. (Which would be caused by a body being killed but NOT removing itself from that array first.) => **might be fixed**

## 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.)

## 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.
* Add “probabilities” to the different elements, allowing me to make more useful/fun ones appear *more often*. (Stuff like the “bomb” is very weird and special, so it should be unlikely to appear.)
* Forbid placing the finish until *all things we want to teach* have been taught?
* Whenever I do something to a jump normal, show a tiny line for that. (Similar to debug, but prettier.)
* Instead of slicing always creating *triangles*, just create two copies of *the old shape*. (Or the old shape – 1.)
* “Coin lock” => perhaps too many coins, that appear too quickly
* Remove *modules* from player bodies that don’t need them.
* On finish, you transform back into your *original shape*. (Helps remember who you are when you see the final standings + nice wrap-up.)
  + Use the “soft lock” mechanism from locks here as well, so you never leave the finish again.
* 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?)

## Optional & Questions

* BUG: Buttons can be placed outside the actual room => make “tiles\_inside” tighter to the inside on locks ( = only use the shrunk rect) **IMPLEMENTED** => does this work and is this a good idea?

# 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.