Ahex Notes

Introduction

Ahex is a minimalistic puzzle game with hexagonal movements. This document lists thoughts and ideas.

Aesthetic

- ☑ I'd like to go for a soft pastel aesthetic, inspired by games such as [Tunic](#).
- ☑ The world has a baseline made of water, which is implemented using the `bevy_water` crate.
- ☑ Tiles are hexagonal with the tip pointing up. This is a stylistic design choice. The hexagon is defined through a circumcircle with unit radius, yielding the following geometry:
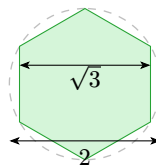

Figure 1: Geometry of a hex tile

Due to Bevy, the *xz*-plane is the "flat" ground plane. The *y* axis points up. Hence, Figure 1 uses the *x* and *z* axes.

Mechanics

- ☑ The camera can rotate, so that the player can see behind tall objects.
- ☑ The camera should rotate only in intervals of $\frac{\pi}{3}$ radians at a time, so that the hexagons always end up looking the same. Of course this transition should be fluent.
- ☐ The controls (`W/E/A/D/Z/X`) should adapt based on the angle of the camera. Otherwise controls are too confusing for the player if the camera is rotated.
- ☐ The player should be able to restart a level using some button, remote from the usual controls. It could be a combination like `Ctrl+R`.
  - ☐ If a level has become unwinnable (due to the player or the *star* falling into the water), the game should hint to use this restart combination.

Puzzle ideas

- ☐ The objective is a *star*. Upon collecting the *star*, the level is completed.
  - ☐ Levels may be replayable by implementing a secondary *star* after completion. I'm not sure yet if I want to do this for every level.
  - ☐ A *star* is subject to physics just like the player. Hence, a *star* can fall down or be lost to the abyss.
- ☑ The player can fall *down*, but not jump *up*. This causes a significant asymmetry for the *y* axis.
- ☐ The player has a certain height. This disallows him from squeezing between two tiles (one above the other) if there isn't enough height left.
- ☐ Tiles can be programmed to move along a *path*. This can have multiple sub-variants:

- [ ] Back and forth between two coordinates: this is useful for simple elevators (going up and down the *y* axis) or short hops to form bridges.
- [ ] A line segment: an extension of just moving back and forth; specify a direction and an amplitude.
- [ ] A circle: to recreate floating platforms that can take the player to multiple places, or to even simulate conveyor belts.
- [ ] The full solution: a directional path, i.e. a `Vec`<(`isize`, `isize`, `isize`)>. This would allow a tile to move multiple coordinates in one step, as well as take any arbitrary path. Of course this path *should* return to the tile's original position, although this is not a strict requirement.
- [ ] If the player is on a tile that is moving, the player should move along with it, keeping in mind collisions.
- [ ] Tiles may be *slippery*. If the player moves on them, the player will continue to move until an end is reached (wall, or edge of the map).
- [ ] Tiles may be *fragile*. After the player has stepped on it, it will crumble as soon as the player steps off it.
  - [ ] Some *fragile* tiles might be rechargable.
- [ ] *Crates* are solid objects that the player can't traverse through, but can push. A push is only possible if the crate can occupy the target hex.
- [ ] The player is only strong enough to push one *crate* at a time (I think). A series of crates are therefore not pushable in the direction that they form a series in.
- [ ] *Lasers* block the player from moving through them, much like walls. Lasers extend across the entire level, until blocked by something solid.
  - [ ] *Lasers* may be blocked by the player pushing a *crate* into its path.
- [ ] *Trampolines* cause the player or any other solid object to jump one tile. This can be used to cross bridges.
  - [ ] *Trampolines* could come in fixed or in *crate*-like variants (which can be moved). Note that for this, tile heights must be uniform!
  - [ ] If a player falls down flat on top of a *trampoline*, they can no longer move in any direction. This should trigger the restart hint.
  - [ ] Jumps should keep into account collisions. The jump might be canceled halfway if the player would otherwise hit a wall. This could cause the player to fall down early.