# Compilation Instructions (Windows)

javac -cp "..\lib\minim\\*;..\lib\core.jar;." .\\*.java

java -cp "..\lib\core.jar;..\lib\minim\\*;." DontDrown

# Mechanics

* Jump
  + Single ~~or double~~?
  + ~~Press and hold to increase height?~~
  + Timings
    - Climb
    - Hang time
    - Fall
    - Coyote-time (jumping just after going off the edge of a platform)
    - Delayed jump frames (if you try to jump just before touching the ground)
    - Floaty or grounded?
  + Mid-air velocity change?
  + Maintained horizontal velocity from ground movement?
  + ~~Wall jumping?~~
* Water
  + Start immediately?
  + Constant speed
  + Instant death, ~~or breath holding mechanic~~?
    - Instant death in Rainbow Islands
    - Restart level upon death
  + Makes tokens inaccessible
  + Opaque ~~or translucent~~?
* Movement
  + Acceleration
    - Slow = heavy
  + Top speed
  + Deceleration
    - Slow = slippery, hard to control
  + Acceleration and deceleration frames do not need to be equal
  + Direction change time
* Camera
  + Vertical scrolling
  + Move when player in top (and bottom?) third (?) of screen
  + First level, start with player in bottom of screen, to encourage moving upwards
* Platform types
  + Single-side pass through
  + ~~No pass through~~
  + ~~Drop-away~~
  + ~~Spiked~~
  + ~~Mobile (side to side etc)~~
  + ~~Slippery~~
  + Colour-coding
  + Trampolines
  + ~~Are the sides of the screens walls that can be jumped off/climbed?~~ 
    - ~~Mario, Super Meat Boy, Celeste~~
* De-buffs
  + Loss of colour contrast (makes ~~distinguishing platforms~~ tracking stress difficult)
  + Tunnel vision, but only stress when water within sight
  + Constant stress
  + ~~Powerups increase stress~~
  + Unpredictable stress spikes
  + Become sluggish when not stressed
  + How should de-buffs be communicated to the player?
    - Descriptions are a bit boring, but they may not be intuitive to understand
    - ~~Tutorial level~~
    - Visual or ~~auditory~~ feedback
* ~~Powerups~~ 
  + Temporarily stop the water, but it speeds up afterwards
  + Cap the impact of stress
  + Temporarily reduce the impact of stress
  + Higher jumps
  + In-level pickups, or between-level purchases?
    - In-level (levels should be re-playable for completionism, the same each attempt)
* Stress effects
  + Jerkier movement
    - Decreased acceleration time
    - Increased deceleration time
    - ~~Reduced hang time~~ 
      * Could increase rise and fall time to maintain overall airtime
  + Faster music
  + ~~Vignette effect~~
  + ~~Directly proportional to how close the water is~~, or increasing/decreasing over time when the distance is within certain thresholds?
    - The former could be quite jerky, but easier to understand
* Difficulty curve
  + ~~Introduction of new platform types~~
  + Introduction of new de-buffs (and powerups)
  + Increased water speed
  + Greater density of tokens in a horizontal space
  + ~~How close platforms are to one another (i.e. platforming difficulty)~~
  + Are levels grouped into “worlds”?
    - By de-buff?
    - ~~By platforming difficulty?~~

# Implementation

* Ground detection
  + Update Boolean upon landing/jumping/going off an edge
  + ~~Check proximity when attempting to move/jump?~~
* Physics engine
  + Force resolution
    - Potentially makes controlling climb, hang time and fall time harder
      * Use different gravity values for rise and fall – they can’t be too different, or it’ll feel wrong
      * Finite state machine of movement
  + ~~Animation-based movements~~ 
    - Makes maintaining horizontal velocity implausible
* Controls
  + WASD
  + Arrow keys
  + Space to jump
  + Pause menu
  + Menu interactions
    - Mouse based
    - Keyboard based

Plays it until it is boring: 15-60 minutes

Table

Description automatically generatedA picture containing text, receipt, screenshot, document

Description automatically generatedTable

Description automatically generated with low confidence