

Topics/Awards/Prizes Link:

https://docs.google.com/spreadsheets/d/1k3ixfYsnp2EPLkSNM7fN8u_nXaNfnz93pjqpvgnzDDk/edit#gid=0

Gamedev:

-Some kind of rhythm game?

-Some topics:

-2D platformer is very safe, but too safe.

-2d top down I've done a lot and is also relatively safe choice

Game Idea One:

- You can jump in stage 1 for the level but after completing it you would be unable to and could only swim or smth in that **SAME EXACT LEVEL** design.
- A singular level, but everytime you complete it there would be new challenges towards it.

Game Idea Two: 2.5D platformer

- 2D player stuck in 3D world, things pop into your plane of existence/push you into a different plane, some kind of platformer revolving around this.

Game Idea Three:

- Super mario galaxy but it's 2D.
- Explore different planets idk

Game Idea Four:

- Have one single level, procedurally generated. A person from another computer can affect how the level goes as you play through their webcam.

Game Idea Five:

- 2D Metroidvania that is based around the effects of coronavirus. (Enter a world where coronavirus has evolved and affected all organisms - kinda like Hollow Knight)

Game Idea Six:

-

Funniest Dev Hack:

- Mock some pre-existing libraries that exist online and create a website to download it.
 - One person can be designing the website while the other goes haywire with coding.
 - E.g. "Legstrap" - essentially a bootleg bootstrap filled with terrible css files, javascript functions and much more.
 - Bruh.js, I swear to god there's so many different javascript libraries that I would believe this would exist too. (react, angular, electron, etc.)

Best Health Hack:

- I was thinking of making some kind of informative game related to covid, but not really sure how it would work out.
- History of COVID-19. (Essentially, take different types of coronaviruses that lead to different stages - we could search up specific timeline events.)
 - The only issue would be assets and the fact that it's more like an interactive slideshow than a game.

Social Impact Hack:

- Was thinking about making some kind of online game to help people socialize

Other Ideas:

Stuff related to OpenCV:

- Draw a platforming level on paper, and then turn it into a game using OpenCV
- Use your hand to make certain shapes, get points on how quickly you get them.

Random things:

- Straight up make a game engine.

Game Idea: Puppeteer (Idk title name needs work)

- There are two players in the game.
- There are two different types of players in the game that constantly swap after completing each level. If a player fails to complete a level, the opposing player wins.

-First Type of Player: **Puppeteer**

- Gets a # of "points" to spend on either...

- Stage Hazards
- Enemies/Monsters
- What ability the other player has.

-If the puppeteer fails to win after designing his level he loses

-Second Type of Player: **Puppet**

- Continues the game if the player survives/completes the level before time is up. (Causes places to swap)

- Is given an ability by the puppeteer and the puppet will have to adapt to this for the level.

Some examples include....**(Create an Ability Class → Will be part of Player Class)**

- (Normal) → Can only walk on ground tiles and swim in water tiles.
- (Amphibian) → Allowed to go on water tiles but can only stay in ground tiles for 5 seconds before dying.
- (Demonic) → Allowed to go on lava but stepping in water kills you.
- (Delusional) → Key controls for movement are switched.
- (Stressed) → Reduced time limit to complete a level
- Random idea: Puppet literally is a puppet

Puppeteer gets to control the player for an uncertain amount of time while the puppet is attempting to pass. The puppeteer can put .5x speed in any

direction. For example, if the puppeteer presses left and the puppet goes right, it only goes right at .5 speed.

When loading data, check if txt exists, if not, convert png to txt, then put data in txt into game.

Tile Structure(1a.png)->1a.txt(formatted as csv):

- Brown: Ground(1)
- Blue: Water(4)
- Red: Lava(3)
- Green: Exit(5)
- Black: Wall (Can't walk into)(2)
- White: Empty space(0)

Other data(as separate 1b.png file)->1b.txt(formatted as csv):

- Player spawn location-Green(1)
- Enemy spawn location-Red(2)
- Empty(0)

Tasks(in order of importance):

- **Logic System(2-3h):**
 - **GameObject/Scene(mid level)/Handler(top level) game loop**
- **Tile Class(30 min)**
- **Input class(30 min)**
- **Draw Demo level in paint.(10min)//Note:these two can be skipped, and go straight to editor.**
- **Img->LevelData converter(15min)**
 - **Level Loader holds tile prototypes.(30min)**
 - More info on levels:
 - Most screens are 1920x1080. (60x30Tiles,32px wide each)
 - Format:"Level_a:[[3,43,43,42,34,23,],[3,43,43,42,34,23,]]||||(Monster,3,4)
 - First Array: Data for Tiles, Second Array:
- **AABBbox(15min)**
- **Button Class(15 min)**
- **Level Editor(3h):**
 - **WASD move cursor 1 tile. Mouse cursor, space to apply,,arrow keys to select tile.**

~8hours have passed.

- **If the level loads edited level and is correct visually:**
- **Physics Class (probably just hitbox class)(1h)**
- **Player Class(2hrs)**
 - **Health,Movement, Collision**
- **Put player in test level, stop project when touching exit, test collision.(1hr)**
- **Put Everything into 'play' scene.(1hr)**
- **Main Menu scene(15min)**
- **Test Main Menu -> LevelLoad(default level 1)->Gameplay(5min)**

- Test Main Menu -> LevelEdit(default level 1)->Gameplay(5min)
- Create Water/Lava/Other Tiles
- Game Select Menu
- Networking API(1hr)
- Adding Image Parameters for Tile.
- Enemies(2hrs)
- End Game Scenes
- Classes/Interfaces:
 - Tile
 - Subject, Observer for checking if requirements passed(ex: don't touch water)
 - Enemies
 - Pathfinding util class
 - Button(buttons you can click)
 - InputHandler(maps buttons to movement)
 - Ability

Flowchart of the Game:

- 1) Main Menu:
 - a) Instructions Button, Puppeteer Button and Puppet Start Button
 - i) Instructions Button → Shows an Image of Instructions, Key Binds, Which buttons to press on the Main-Menu, (Could add GIFS),
 - ii) Clicking on the Puppeteer Start Button makes the program user be the server.
 - iii) Clicking on the Puppet Start Button makes the program user be the client.
- 2) When both players have connected (Communicate with each other), The Puppeteer begins spending points to use for making the map. There is one default map for users to play.
 - a) Once the Puppeteer is done creating, he will test the map.
 - b) If the Puppeteer fails to complete the map he made, **Defeat** screen for the Puppeteer and **Victory** for the Puppet
 - c) If the Puppeteer does complete the map he made, pass the map data to the Puppet and the Puppet plays the game.
- 3) If the Puppet fails to get to the end of the game, **Defeat** screen for the Puppet and **Victory** for the Puppeteer.
 - a) However, if he/she succeeds, then roles **swap**.
 - b) Repeat steps 2-3 indefinitely until someone fails to complete the level.

Scenes:

- 1) Main Menu
- 2) Instruction Page
- 3) Puppeteer creating the map
- 4) Puppeteer playing the map
- 5) Puppet playing the map