Godot Notes

**Syntax**

* Similar to python
* Variables types can be inferred using : = or declared using the : operator.
* Pass can be used in functions where code isn’t yet written
* For Boolean comparisons, both or and || are accepted, same with other comparison operators.

**\_physics\_process()**

* Called every frame
* Comparable to fixed update in unity.
* Use for everything to do with collision
* Physics processes in sub classes will be run after the physics process in the main class.
  + Eg if player extend s actor and they both have a physics process, actors will be run first and players after.

**\_ready()**

* Godot equivalent of Start()
* Gets called from top to bottom from the scene tree, when it hits a node that has children it will call the bottom child first and work its way up to the top of the node.

**Setting Variable Types**

* In Godot, to set a variable type you declare the variable as var
* You then use the : = operator to set the type
* Eg. var velocity: = Vector2(1,1)

**Making variables modifiable in the inspector**

* Use export keyword in front of variable to show in inspector
* Eg export var gravity : = 3000

**Tile Map**

* Can use tile map to build 2D levels
* Can import png tile map and split up in Godot, can also apply collisions

**Layers**

* In a nodes collision tab, can select which physics layer the object is
* New layers can be added in the project settings
* Masks can be created in the collisions tab to decide what an object should interact with, settings these up correctly can improve performance.

**Input**

* Input map can be found in project settings, can assign button presses to labels to be used in code.
* Using get\_action\_strength returns a float for the input value
* Is\_action\_just\_pressed is like GetKeyDown in unity.

**Godot Kinematic Functions**

* Only get called after position is updated
* Is\_on\_floor() is a kinematic2D function that checks if the object is on floor but needs floor normal set in the move\_and\_slide function.

**Tertiary Operator**

* Instead of test1 = test1 < 2 ? true : false
* Have test1 = 1 if input < 1 else 1

**Time**

* Can use get\_physics\_process\_delta\_time() to get time similar to Time.deltaTime in unity.

**Editor**

* If you want to rename a variable and change all instances of its name can press Ctrl + R to open up the replace function.
* Find all references to a variable is Ctrl + Shift + F

**VisibilityEnabler2D**

* A node that can be added to 2D nodes that will stop scripts running on a node when outside of the view.
* There is a checkbox for Physics Process Parent that when checked will not run the physics process on a node when it is out of view.
* Set\_physics\_process can be used to stop all processes until the enemy is in view.

**Queue\_free()**

* Will delete node.

**Get\_node()**

* Equivalent to get component.

Things to Learn Before Starting

* ~~Basic Tutorial for Platformer~~
* ~~2D lighting~~
* 2D animations
* 2D particles
* User Interface
* Parallax backgrounds
* Audio System
* Input/Output of Files
* Mobile Development

Things to Play Around With For Project

* Godot Midi parsing
* Parsing guitar hero charts
* MP3 support through libraries
  + <https://github.com/MightyPrinny/godot-FLMusicLib>
  + https://github.com/DeleteSystem32/godot-minimp3
  + https://github.com/lieff/minimp3
* Use code from audio visualizer to get lights to pulse to music
  + <https://www.youtube.com/watch?v=AwgSICbGxJM>
* Tools for perfect audio sync
  + https://docs.godotengine.org/en/stable/tutorials/audio/sync\_with\_audio.html

# AI Idea

When AI is out of view totally stop running all AI processes, when the player is almost able to see the AI, quickly calculate based on the AI’s speed and the players current position where they would be if they had have kept moving and move them there