Game Technologies

Semester project
Code Name: Platformer

Romain Claret, Quentin Vaucher

https://github.com/Penguinazor/mse.gt.project

Table of contents

1. Introduction
2. Gameplay
3. Network
4. Artificial Intelligence
5. Ending a game
6. Demo
7. Improvements
8. Conclusion

Introduction

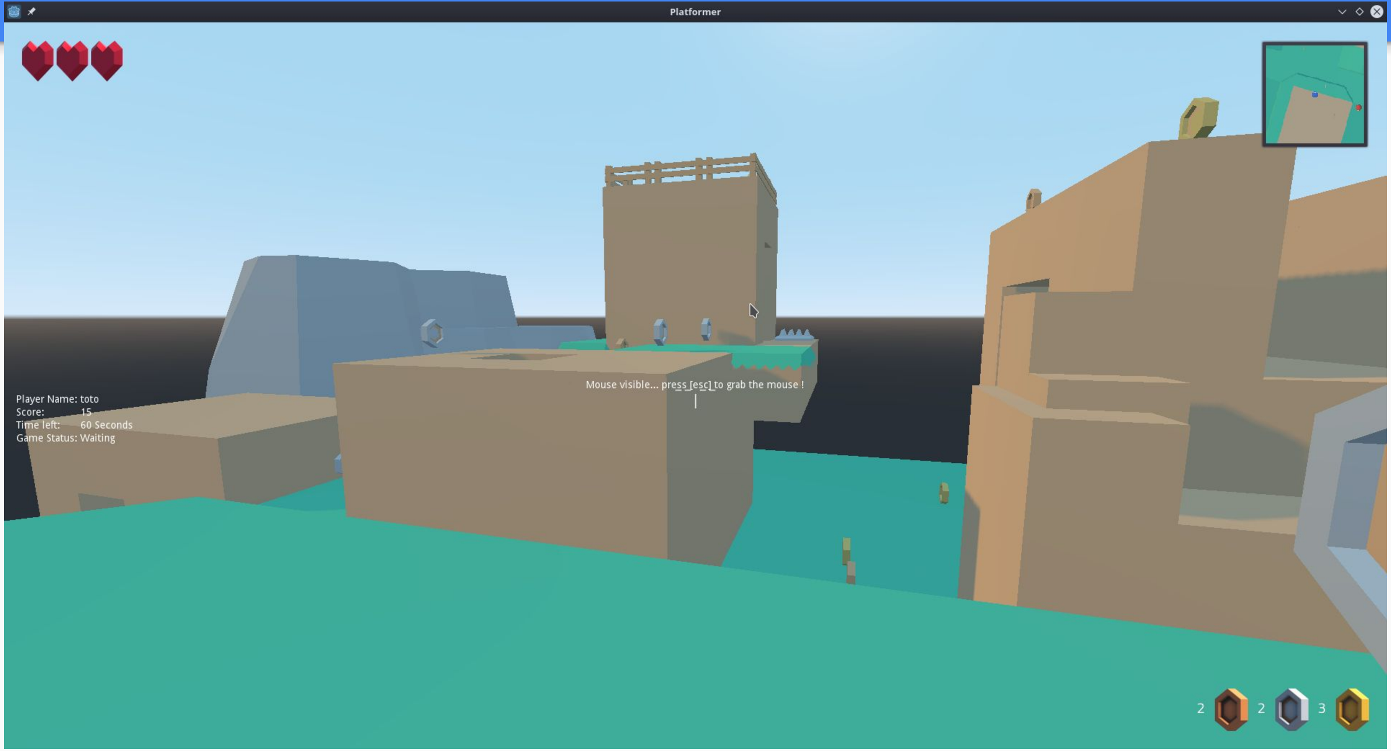
- Godot Game Engine
- Open Source, multiplatform
- GDScript, Python Superset
- Young Game Engine
  - Started in 2007, open sourced in 2014
  - Unity was released in 2005
- Good opportunity to learn something different
- Challenging

Gameplay

- Heads-Up Display
- Map
- Collectables
- Bot

Gameplay - HUD

- 1st/3rd View
- Minimap
- Lifes
- Score
- Collectibles
- Time left
- Status
- Flashlight
- Running
- <esc>



Player Name: toto
Score: 15
Time left: 60 Seconds
Game Status: Waiting

Mouse visible... press [esc] to grab the mouse !

Gameplay - Map

- Possibility to use the environment
  - Platforms (jumps, …)
  - Houses, blocks (hide)
- Interactions
  - Lift
  - Spikes

Gameplay - Collectables

- Collectible by collision
- Pieces are ray-casted randomly on the map
- Different values:
    - Bronze: 1pts
    - Silver: 2pts
    - Gold: 3pts

Gameplay - Bot

- To distract the players
- One NPC on the map
- Moves across the map
- Shots at players
- Immortal
- Bot temporarily freezes on hit

Network

- Client / Server architecture
- Sharing information is done using *Remote Procedure Call*
    - RPC
    - Sync RPC
    - Master/Pupper RPC
- Multiplayer API has a rich abstraction layer
- Godot Debugger attached only to one instance

Artificial Intelligence

Three different states:

1. IDLE: Waiting at initial position, looking for enemies in the neighborhood
2. Hunt: Following a player, and attack when it is close enough
3. Retreat: Going back to initial position

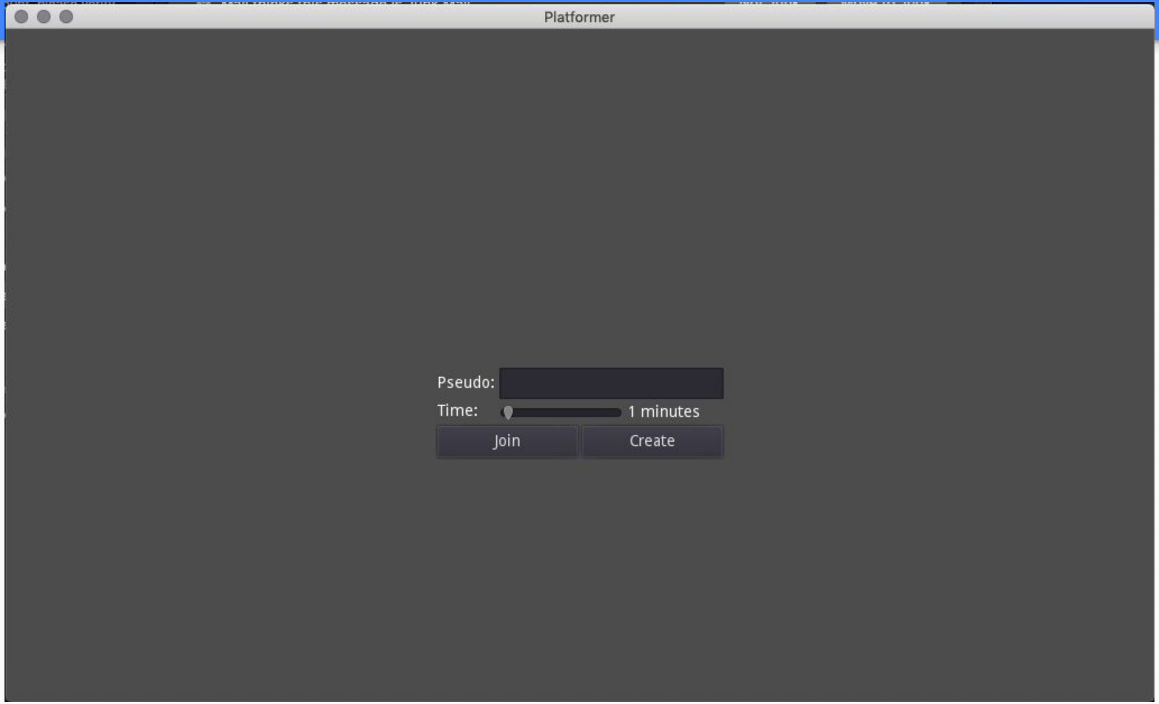
Inflicting damages to the Bot stuns it for three seconds

Ending a game

Two different endings are possible:

- Killing the opposite player
- Highest score at the end of the allocated time

Demo



Pseudo:

Time: 1 minutes

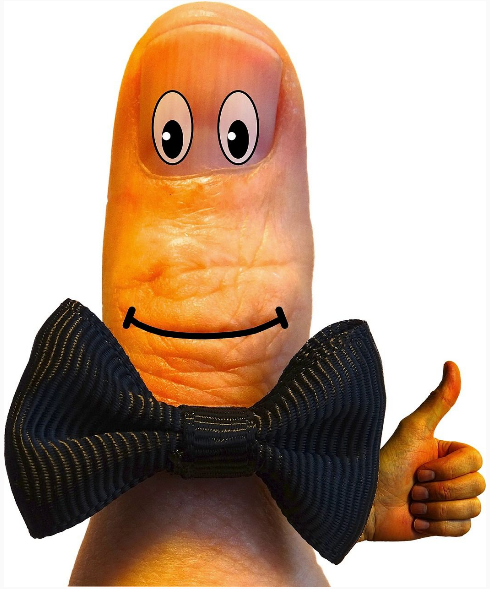
Improvements

- Map improvements:
  - Spikes
  - More interaction
  - Close borders
- Unfinished Features / Bugs
  - Need to restart the server to restart a game session
  - Player dies if falling out of edge
- Gameplay:
  - Different type of Snowballs

Conclusion

- It is a good prototype but not a complete game
- Using Godot was a good experience, but…
  - Lack of documentation/examples
  - Lack of community questions online
- Constantly evolving APIs

Questions ?



Checklist

- Terrain
- Personnage / Mouvement
- Ressources : Internes – Importées / Material – Texture
- Génération aléatoire des Ressources et placement aléatoire
- Gestion de 3 caméras : FPS – 3rd – MiniMap
- Lumière directionnelle
- Lumières ponctuelles décoratives
- Collection des « Ressources » avec les collisions
- Permettre au personnage de lancer des projectiles en utilisant des forces
- Les projectiles sont aussi des colliders avec un timeout de vie
- Les projectiles explosent en cas de collision avec un système à particules
- Des objets qui tombent du ciel avec un Raycast pour la détection du sol
- Initier des composants UI pour afficher les indicateurs du jeu
- 2 Joueurs se confrontent sur le même terrain
- Le jeu se termine si un joueur tue un autre (projectiles et barre de vie)
- Le jeu se termine après une durée maximale (score selon ressources accumulées)
- Network: Gestion des positions des joueurs et des ressources
- Gameplay: projectiles et des collisions des projectiles et des barres de vies
- Gameplay: Gestion de la récupération des ressources et des scores
- AI Déplacement au sein de la scène
- AI Comportement : attendre, attaquer, revenir