Tanks

For Programming Challenge

N. H. N. D. de Silva
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Introduction

- Cross domain projects
  - Software engineering
  - Databases
  - Networking
  - Graphics
  - AI!
- Same Game so tougher grading
- The LMS will be available....Soon
Introduction
Introduction
Introduction
Introduction
Introduction

• The game objective is; clients acting as “tanks” accumulating points while making sure of their own survival.

• The tanks are capable of shooting shells (bullets) and these bullets will move 4 times faster than a tank.

• The environment is made out of four kinds of blocks
  • Brick wall
  • Stone wall
  • Water
  • Blank cell
Game in brief

- Introduction
- **General Game Architecture**
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
General Game Architecture

- Server
- Client 1
- Client 2
- Client 3
- Client 4
- Client 5
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Game Initiation

- A timer gets initiated when the first player registers for a game.
- When the 5th player joins the game or in the event of the timer expiring, the server will issue the game starting message to all the players that are already registered with the server.
- As discussed before, any join request by a client hereafter until the end of the game instance will be rejected.
- The game will be played in a 20x20 grid.
Server Architecture

UI layer

Game Engine

Network Layer
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Joining the game

- The client will have to send a request to the Server saying `JOIN#`
- The server will reply with one of the following replies:
  - Join accept message (explained in the next slide)
  - `PLAYERS_FULL#` - The maximum number of players are already added
  - `ALREADY_ADDED#` - The player is already added
  - `GAME_ALREADY_STARTED#` - The player tries to join an already started game
Joining the game – Acceptance

- If the client’s request can be honoured, the server replies with a message in the following format.

  \[ \text{S:P<num>: <player location x>,<player location y>:<Direction>#} \]

- For an example the first player to register in a game instance may get a message like,

  \[ \text{S:P1: 1,1:0#} \]

- Direction
  - 0 North
  - 1 East,
  - 2 South
  - 3 West
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- **Game initiation**
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Game Initiation

- All other map details vary with each game instance and will be sent in the following format:

  I:P<num>: <x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>......<x>,<y>#
Game Initiation

- All other map details vary with each game instance and will be sent in the following format:

  I:P<num>: <x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>: <
  x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>: <x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>;
  x>,<y>......<x>,<y>#

- The highlighted part will be your player name. Compare it with what you got for the S message.
All other map details vary with each game instance and will be sent in the following format:

I:P1: <x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>: <x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>;

The highlighted part will have the Brick co-ordinates.
Game Initiation

- All other map details vary with each game instance and will be sent in the following format

I:P1: <x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>: <x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>: <x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>;

- The highlighted part will have the Stone co-ordinates
Game Initiation

- All other map details vary with each game instance and will be sent in the following format:

I:P1: <x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>:<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>: <x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>;<x>,<y>......<x>,<y>#

- The highlighted part will have the Water co-ordinates
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
When the client want to move/rotate their tank or shoot, it must issue the relevant command from the following list.

- **UP**#
- **DOWN**#
- **RIGHT**#
- **LEFT**#
- **SHOOT**#

A client can issue one of these commands each second.
Moving and Shooting

- If the request direction is the same as the current direction, the server will try to move the client 1 cell in the said direction.
- Otherwise the tank will be rotated to face the requested direction

**E.g.:**

- If the tank is facing **North** and the client issues the command **UP#**, the server will try to move the tank up by 1 cell
- If the tank is facing **North** and the client issues the command **DOWN#**, the server change the tank direction to **South**.
Moving and Shooting

- If the request of the client cannot be honoured due to some reason, it will be immediately communicated.
  - OBSTACLE#
  - CELL_OCCUPIED#
  - DEAD#
  - TOO_QUICK#
  - INVALID_CELL#
  - GAME_HAS_FINISHED#
  - GAME_NOT_STARTED_YET#
  - NOT_A_VALID_CONTESTANT#
- Otherwise the server will update its internal variables
Moving and Shooting

- Once per second the server will broadcast a global update to all the clients documenting its internal variables about the clients.

G:P1; <player location x>, <player location y>, <Direction>, <whether shot>, <health>, <coins>, <points>: ....
P5; <player location x>, <player location y>, <Direction>, <whether shot>, <health>, <coins>, <points>: <x>, <y>, <damage-level>; <x>, <y>, <damage-level>; <x>, <y>, <damage-level>; <x>, <y>, <damage-level>..... <x>, <y>, <damage-level>#
Moving and Shooting

G:P1;<player location x>,<player location y>;<Direction>;<whether shot>;<health>;<coins>;<points>: .... :P5;<player location x>,<player location y>;<Direction>;<whether shot>;<health>;<coins>;<points>: <x>,<y>,<damage-level>;<x>,<y>,<damage-level>;<x>,<y>,<damage-level>;<x>,<y>,<damage-level>;<x>,<y>,<damage-level>.....<x>,<y>,<damage-level>#

- The highlighted part will have the state data for player 1
Moving and Shooting

G:P1;<player location x>,<player location y>;<Direction>;<whether shot>;<health>;<coins>;<points>: .... :P5;<player location x>,<player location y>;<Direction>;<whether shot>;<health>;<coins>;<points>: <x>,<y>,<damage-level>;<x>,<y>,<damage-level>;<x>,<y>,<damage-level>;<x>,<y>,<damage-level>;<x>,<y>,<damage-level>;<x>,<y>,<damage-level>.....;<x>,<y>,<damage-level>#

- The highlighted part will have the state data for player 5
Moving and Shooting

G:P1;< player location x>,< player location y>;<Direction>;< whether shot>;<health>;< coins>;< points>: .... :P5;< player location x>,< player location y>;<Direction>;< whether shot>;<health>;< coins>;< points>: < x>,<y>,<damage-level>;< x>,<y>,<damage-level>;< x>,<y>,<damage-level>;< x>,<y>,<damage-level>;< x>,<y>,<damage-level>......< x>,<y>,<damage-level>#

0 (no damage) , 1 (25% damage), 2 (50% damage), 3 (75% Damage), 4 (100% Damage).
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Acquiring coins

- For two reasons coins will appear on the map.
  - Random treasures
  - Spoils of war

- The players are supposed to collect them by moving to the cell where the coin pile is.

- When a player collects a pile of coins, their coin count and the point count will get increased by that amount.
The server will send the following message to signal the appearing event of a pile of coins.

C:<x>,<y>:<LT>:<Val>#

- x - <int>
- y - <int>
- LT - <int> the time that the pile of coins will be on the map before it automatically disappears
- Val - <int> value of the coins
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Acquiring points

- There are three ways to acquire points
  - **Breaking bricks**
    - Each time a shell (bullet) from a player damages a brick wall, a constant amount of points is added to the player’s point count.
  - **Collecting coins**
    - As described in the previous topic
  - **Spoils of war**
    - Will be discussed in a subsequent topic
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Life Packs

• Similar to the piles of coins, life packs will appear on empty cells.
• The method of taking a life pack is similar to the method of taking a treasure.
• When a client takes a life pack, 20% of the initial health will be added to their health.
• Note that it is possible for a client, at some point of the game, to have a health value which is greater than the initial health value.
Life Packs

- The server will send the following message to signal the appearing event of a Life Pack.

  \[ L: <x>, <y>: <LT> \]

  - \( x \) - \(<\text{int}>\)
  - \( y \) - \(<\text{int}>\)
  - \( LT \) - \(<\text{int}>\) the time that the life pack will be on the map before it automatically disappears
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Killing!

- A bullet hit will reduce the health of a tank by 10% of the initial health.
- A player who has health of 0% (or less) will die and drop their coins at the point of death. But they will not lose the current points. However, 25% of the points of the victim will be added to the killer.
- The dropped pile of coins is no different from a randomly created pile of coins. Thus the killer do not have a claim on it, it can be picked up by any remaining client.
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Penalties and Death

- If a client hits an obstacle (a standing brick wall or a stone wall), a fixed number of points will be deducted from the client.
- The deduction takes place per each turn, thus if a client keeps hitting an obstacle, it will continuously lose points.
Penalties and Death

- As described in a previous slide, clients can shoot over water. But if the client tries to move the tank into a cell with water, the client will be killed.
- Since there is no killer here other than the environment, no body will be credited with points and the coins of the slain client will not drop to the ground.
- But they will get their coin count set to zero nevertheless.
Game in brief

- Introduction
- General Game Architecture
- Server Architecture
- Joining the game
- Game initiation
- Moving and Shooting
- Acquiring coins
- Acquiring points
- Life Packs
- Killing!
- Penalties and Death
- Endgame
Endgame

• The game ends if one of the following happens;
  • All the clients are dead
  • The game lifetime counter expires
  • All the coin piles (random or otherwise) finished appearing on the map and all the appeared coin piles either has been claimed by a client or disappeared.

• When the game ends, the player with most points will be declared the winner. And the message **GAME_FINISHED** will be broadcasted to all the clients.
Q & A
Thank you!

“Live long and prosper”