

Tetris Battles 2 Player!

Shaeq Ahmed, Anthony Liang, Brian Yang

January 24, 2017

Abstract

This project is intended to be a replica of the popular Facebook game Tetris Battle, in which players play Tetris against each other and can add lines to their opponent's game by completing lines in their own game. Players can also see their opponents playing at the same time on their screens. In this version of Tetris Battle, the terminal will be used as the GUI. Beyond basic gameplay, a core focus of our Tetris game is on asynchronous gameplay (pseudo multiplayer). Our version of Tetris uses the standard grid size, a board 10 units wide and 20 units high. The game will have seven color coded tetromino shapes which can be represented with letters that suit the overall structure of the piece: I, J, L, O, S, T, Z.

1 Function Headers for Server.C

```
int main();
```

Inputs:None

Returns: 0

Explanation: Starts and runs the server, forking off subserver to communicate with clients

```
void subserver1();
```

Inputs: int connection, int read_pipe, int write_pipe, char buffer

Returns: None

Explanation: Starts a subserver to communicate between the other subserver and one of the clients

```
void subserver2();
```

Inputs: int connection, int read_pipe, int write_pipe, char buffer

Returns: None

Explanation: Starts a subserver to communicate between the other subserver and one of the clients

2 Function Headers for Client.C

```
int main();
```

Inputs:None

Returns: 0

Explanation: Runs the client, starting the game and communicating with the server at the same time

3 Function Headers for Networking.C

```
void error_check();
```

Inputs: int i, char s

Returns: None

Explanation: Checks for errors

```
int server_setup();
```

Inputs: None

Returns: int sd (file descriptor of server)

Explanation: Setups the server

```
int server_connect();
```

Inputs: int sd

Returns: int connection

Explanation: Connects the server with the client

```
int Client_connect();
```

Inputs: char* host

Returns: int sd

Explanation: Connects the client with the server

4 Function Headers for Tetris.C

```
void ff_move(game *obj, int direction);
```

Inputs: game *obj, int direction

Returns: None

Explanation: Left and right movement for tetromino.

```
void ff_init(game *obj, int rows, int cols);
```

Inputs: game *obj, int rows, int cols

Returns: None

Explanation: Initial setup of variables and the game object.

```
void ff_get_line(game *obj, int r, WINDOW board);
```

Inputs: game *obj, int r, WINDOW *board

Returns: None

Explanation: Shift board upwards and add a "sent" line at the bottom of the board with a bomb placed at a random location.

```
game* ff_create(int rows, int cols);
```

Inputs: int rows, int cols

Returns: game *

Explanation: Create the game object and allocate memory of it.

```
void ff_destroy(game *obj);
```

Inputs: game *obj

Returns: None

Explanation: Free the board when game is over.

```
void ff_delete(game *obj);
```

Inputs: game *obj

Returns: None

Explanation: Free the game object itself.

```
void ff_remove(game *obj, piece piece);
```

Inputs: game *obj, piece piece

Returns: None

Explanation: Clear a row from the board.

```
void ff_fits(game *obj, piece piece);
```

Inputs: game *obj, piece piece

Returns: None

Explanation: Check if piece fits on the board.

```
void ff_down(game *obj);
```

Inputs: game *obj

Returns: None

Explanation: Drop a tetromino piece to bottom.

```
static void ff_new_falling(game *obj);
```

Inputs: game obj

Returns: None

Explanation: Initialize new falling piece and populate the next piece with a random tetromino.

```
void ff_hold(game *obj);
```

Inputs: game *obj

Returns: None

Explanation: Swap the falling piece with the piece in the hold buffer.

```
void ff_rotate(game *obj, int direction);
```

Inputs: game *obj, int direction

Returns: None

Explanation: Rotate piece.

```
char ff_get(game *obj, int row, int col);
```

Inputs: game *obj, int row, int col

Returns: char

Explanation: Return the piece at the given row and column.

```
static void ff_set(game *obj, int row, int col, char value);
```

Inputs: game *obj, int row, int col, char value

Returns: None

Explanation: Set the piece at the given row and column.

```
bool ff_check(game *obj, int row, int col);
```

Inputs: game *obj, int row, int col

Returns: boolean

Explanation: Boundary checking and collision detection.

```
static void ff_put(game *obj, piece piece);
```

Inputs: game *obj, piece piece

Returns: None

Explanation: Place a piece onto the board.

```
void ff_tick(game *obj);
```

Inputs: game *obj

Returns: boolean

Explanation: Do a single game tick: process gravity, user input, and updates score.

```
static bool ff_line_full(game *obj, int i);
```

Inputs: game *obj, int i

Returns: boolean

Explanation: Returns true if line i is full.

```
static void ff_shift_line(game *obj, int r);
```

Inputs: game *obj, int r

Returns: None

Explanation: Shift every row above r down one.

```
static int ff_check_line(game *obj);
```

Inputs: game *obj

Returns: int

Explanation: Find rows that are filled, remove them, shift the board down, and return the number of cleared rows.

```
static void ff_adjust_score(game *obj, int lines_cleared);
```

Inputs: game *obj, int lines_cleared

Returns: int

Explanation: Adjust the score based on how many lines you cleared at once. Combo clears give more points.

```
static bool ff_game_over(game *obj);
```

Inputs: game *obj

Returns: bool

Explanation: Return true if the game is over.

5 Function Headers for Main.C

```
void display_board(WINDOW*, game* obj);
```

Inputs: WINDOW* w, game *obj

Returns: None

Explanation: Prints entire tetris board.

```
void display_piece(WINDOW* w, piece piece);
```

Inputs: WINDOW* w, piece piece

Returns: None

Explanation: Displays a single tetris piece

```
void display_score(WINDOW* w, game* ff);
```

Inputs: WINDOW* w, game ff

Returns: None

Explanation: Prints and formats the score in a window.

```
void init_colors();
```

Inputs:None

Returns: None

Explanation: Starts and defines all the color pairs for use in ncurses

```
game* init_game();
```

Inputs:None

Returns: game* ff

Explanation: Makes a new tetris game with all the settings and returns it

```
void end_game();
```

Inputs:None

Returns: None

Explanation: Clears out the tetris game windows, frees the memory.