**Starpusher Technical Manual**

By Jacob Morgan, Kyle Reinholt, and Tyler Thaxon

**I. Preface**

I’d like to start this technical manual by saying we’re sorry. This code is very sloppy. It’s not well separated; it’s all crammed into one file. But hey, it works. We worked hard on it and we got a working product in the end. Given more time we probably could’ve optimized every facet of the project, but if this is all there was, then… “ya did good, kid.”

**II. Table of Contents**

I. Preface

II. Table of Contents

III. Program Structure

i. Classes

ii. Functions

iii. Variables

**III. Program Structure**

i. Classes

Class LTexture

Public:

LTexture() // This is a standard constructor

~LTexture() // This is a standard deconstructor

bool loadFromFile(std::string path ) // Loads an image at the designated path

void free() // Deallocates textures

void setColor( Uint8 red, Uint8 green, Uint8 blue ) // Set color modulation

void setBlendMode( SDL\_BlendMode blending ) // Set blending

void setAlpha( Uint8 alpha ) //Set alpha modulation

void render( int x, int y, SDL\_Rect\* clip = NULL, double angle = 0.0, SDL\_Point\* center = NULL, SDL\_RendererFlip flip = SDL\_FLIP\_NONE ) // Renders texture at given point

int getWidth(), int getHeight()//Gets image dimensions

Private:

SDL\_Texture\* mTexture // The actual hardware texture

int mWidth, int mHeight // Image dimensions

Class Tile

Public:

Tile( int x, int y, int tileType ) // Initializes position and type

void render( SDL\_Rect& camera ) // Shows the tile

int getType() // Get the tile type

SDL\_Rect getBox() // Get the collision box

Private:

SDL\_Rect mBox // The attributes of the tile

int mType // The tile type

Class Dot

Public:

Dot() // Standard constructor

static const int DOT\_WIDTH = 34, static const int DOT\_HEIGHT = 36 // The dimensions of the dot

int handleEvent( SDL\_Event& e , Tile \*tileSet[]) //Takes key presses and adjusts the dot's velocity

void move( Tile \*tiles[], int direction ) // Moves the dot and check collision against tiles

void setCamera( SDL\_Rect& camera ) //Centers the camera over the dot

void render( SDL\_Rect& camera ) // Shows the dot on the screen

int getX (), int getY() // Returns X and Y values of the dot

void setPosition(int, int) // Passes in X and Y coordinates for the dot

Private:

SDL\_Rect mBox // Collision box of the dot

int mVelX, mVelY // The velocity of the dot

Class Star

Public:

static const int DOT\_WIDTH = 34, static const int DOT\_HEIGHT = 36 // The dimensions of the star

Star() // Standard constructor

Star(int, int) // Parameterized constructor

void handleEvent( SDL\_Event& e , Tile \*tileSet[]) // Determines the velocity for movement

int move( Tile \*tiles[], int direction ) //Moves the dot and check collision against tiles

void render( SDL\_Rect& camera ) // Shows the star on the screen

int getX(), int getY() // Returns X and Y values of the star

void setPosition(int, int) // Passes in X and Y coordinates for the star

Private:

SDL\_Rect mBox // Collision box of the star

int mVelX, mVelY // Velocity of the star

Class Goal

Public:

static const int DOT\_WIDTH = 75, static const int DOT\_HEIGHT = 56 //The dimensions of the goal

Goal() // Standard constructor for the class

Goal(int, int) // Parameterized constructor

void render( SDL\_Rect& camera ) // Shows the star on the screen

int getX(), int getY() // Returns X and Y coordinates of the goal

void setActive(int, int) // Sets “true” if occupied by a star, false otherwise

void setOff() // Sets the goal to “false”

bool getActive() // Returns the state of the goal

Private:

SDL\_Rect mBox // Collision box of the dot

bool isActive // The current state of the goal

ii. Functions

void solve() // Displays the “stage solved” window to the screen

void starOnStar(Dot \*dot, Star \*star, Star \*star2, Tile \*tileSet[], int movement ) // Detects if a star is touching another star

void DotOnStar (Dot \*dot, Star \*star, Tile \*tileSet[], int movement ) // Detects and handles movement between the player and a star

bool touchesWall( SDL\_Rect box, Tile\* tiles[] ) // Returns whether the player or the star is touching a wall

iii. Variables

//Screen dimension constants

const int SCREEN\_WIDTH = 675;

const int SCREEN\_HEIGHT = 616;

//The dimensions of the level

const int LEVEL\_WIDTH = 675;

const int LEVEL\_HEIGHT = 616;

//Tile constants

const int TILE\_WIDTH = 80;

const int TILE\_HEIGHT = 80;

const int TOTAL\_TILES = 99;

const int TOTAL\_TILE\_SPRITES = 12;

//The different tile sprites

const int TILE\_RED = 0;

const int TILE\_GREEN = 1;

const int TILE\_BLUE = 2;

const int TILE\_CENTER = 3;

const int TILE\_TOP = 4;

const int TILE\_TOPRIGHT = 5;

const int TILE\_RIGHT = 6;

const int TILE\_BOTTOMRIGHT = 7;

const int TILE\_BOTTOM = 8;

const int TILE\_BOTTOMLEFT = 9;

const int TILE\_LEFT = 10;

const int TILE\_TOPLEFT = 11;