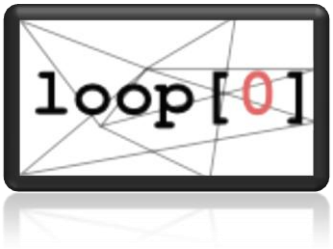

Loophole (Untitled Project Zero)

Final Report

Name: Ryan Hagood

December 8, 2009

Table of Contents

Application Development

Project Description

Loophole is a puzzle game engine created to be structured for educational use and for users to add-on to the code base in the future or implement scripting to create their own game software. The example program implemented will be the game of Sokoban, a Japanese block pushing game.

Similar Applications

There are several projects which are similar to the one that is proposed. The game "Blobbo" is one inspiration for the proposed project which is similar. Both software packages involve using logic to solve puzzles such as learning what steps should be taken and in what order to collect all of the treasure chests in a level and then make it to the exit. The idea of having traps and logical puzzles such as in this game are what will be put into the "Loophole" game project.

Another game which inspiration is drawn from is the game "Chip's Challenge". The basis of "Chip's Challenge" was to go through a maze type level and collect keys to open doors to find the exit. This principle idea will be utilized in the game play of "Loophole" in the progression of the game, primarily the idea of finding the way through a maze area.

A third game which is similar to the system "Loophole" will use is the Japanese game of "Sokoban." "Sokoban" is a block pushing game where the blocks are pushed onto platforms to advance to the next level.

Motivation for Choice of Project

I would like to work on this project because I have always wanted to program a game but usually I disallow myself to become involved in the process to make a game due to the graphical aspect of the programs. A solution to this temporarily could be to make a text based game initially and then expand it into a graphical interface. Game design and programming has always interested me, but I always either think it will be too easy to do something or too hard. This is why I would like to work on this project. After I complete this program I will have gained ground in game programming to help me better work on game programming in the future.

Project Challenges

The challenges anticipated on this project are primarily the concepts of game programming and learning how to use a graphical system such as images to create a game. The level design aspect is also anticipated to create difficulty especially if part of the task is to come up with creative and interesting levels for players to play.

Project Resources

Platform	Linux
Language	C++
IDE	Emacs
Libraries	Simple DirectMedia Layer (SDL)

Websites used http://lazyfoo.net/
SpriteLib sprite collection, licensed under the GPL (http://www.opensource.org/licenses/cpl.php)
Author: Ari Feldman
Level designs from http://www.sourcecode.se/sokoban/levels.php, level set "Novoban"
Author: François Marques

Project Plan

1. Develop a GameState class
2. Develop simple subclasses
3. Make a Level subclass
4. ~~Make a class to control objects in the level.~~ Revised December 7, 2009; removed
5. Design Levels
6. ~~Design a standard interface~~. Revised December 7, 2009; removed

Classes

GameState
- GameInit
- Title
- Level

Timer
~~Objects~~ Revised December 7, 2009; removed

Software Requirements Specification

Functional Requirements

Requirement 01	***Title Screen***
Statement	There will be a title screen for the game.
Evaluation	Title screen is shown upon launching application.
Dependency	None
Priority	Medium
Revision History	December 7, 2009 – Reworded better
FR-01	

Requirement 02	***Levels for Play***
Statement	There will be example levels.
Evaluation	The player can test various levels to see different puzzles.
Dependency	None
Priority	Medium
Revision History	
FR-02	

Requirement 03	***Level Editor***
Statement	Levels may be edited.
Evaluation	User may press a key to access edit mode.
Dependency	FR-02
Priority	Medium
Revision History	December 7, 2009 – Reworded better

FR-03

Requirement 04	*Layers*
Statement	There will be multiple layers in the game's rendering on level
Evaluation	There are multiple layers which can be enabled and disabled
Dependency	FR-02
Priority	Medium
Revision History	Added December 7, 2009

FR-04

Vision and Scope

Vision

Open source software and education from open source software is becoming important in the world of computers. This is seen through projects such as the OLPC (One Laptop Per Child) project, Sugar project, and many company's adoptions of open source software recently. Open source games hold a special ground; they may teach children how to do solve certain puzzles created in the game world as well as show them how the underlying game works. An open source educational game could further allow a teacher or a professor to modify the game to customize it for their lesson plans.

Revised 9/22 from suggestions in mini presentation such as removing "I"

Scope

The project "Loophole" will be a simple puzzle game engine. This game project will be a graphical software package that utilizes logical thinking and ideas to complete specific objectives at various stages. The puzzle game "Sokoban" will be implemented to test the game engine.

Revised 9/22 from suggestions in mini presentation

System Design and Architecture

- GameState.*
  - GameState
    - doEvents — Function used by derived classes to handle events in all states.
    - doLogic — Function used by derived classes to handle logic in all states.
    - doRender — Function used by derived classes to handle rendering in all states.
    - ~GameState — Destructor for GameState
  - setNextState — Sets the next state to newState.
  - changeState — Removes current state and loads up the next state.
  - setPrevState — Sets state to the previous state (unused currently)
- GameInit.*
  - GameInit : public GameState
    - doEvents — see GameState
    - doLogic — see GameState
    - doRender — see GameState
- NewLevel.*
  - Level : public GameState

        - doEvents — see GameState
        - doLogic — see GameState
        - doRender — see GameState
        - RenderBack — Renders back layer of map
        - RenderFront — Renders front layer of map
        - RenderObject — Renders player and objects
        - MovePlayer — Moves player to a different tile on the level
        - CreateBlock — Creates a block on the map specified by the player
        - Finish — Checks to see if player can advance to next level
        - Resize — Resizes the level in edit mode
        - ChangeGround — Changes the ground texture
        - SaveLevel — Saves changes to map
        - Pause — Pauses for a set time period (unused currently)
        - ReturnIndex — Returns index value for a character on map
        - nextLevel — Sets a variable storing next level map name
        - ResetLevel — Reloads the level
- structures.h
    - dim — Dimensions of an object (width and height)
    - coords — Coordinates of an object (x and y)
    - tile — X and Y and tile character of a tile
- Timer.*
    - Timer
        - tickTimer — Increments time and calculates time since last call.
        - getTimeDifference — Return the delta time (time between calls of tickTimer( ))
        - getTime — Returns current time overall since program began.
        - getFramesPerSec… — Returns the frames per second (calculated by 1000/dt)
- Title.*
    - Title : GameState
        - doEvents — See GameState
        - doLogic — See GameState
        - doRender — See GameState
        - loadImage — Loads an image
        - applySurface — Applies one surface to another

About the Author

Ryan Hagood is a student at Berea College whom is extremely interested in open source software, the Free Software Foundation, the Electronic Frontier Foundation, and the Open Source Initiative. He would like to work in an open source company in the future to better the community through shared knowledge and experiences. He can be currently contacted at hagoodr@gmail.com with any ideas on any open source projects he could attempt because of his extreme interest in the subject. He also enjoys hacking (using the traditional term, not the media term). Security is also of interest to him.

Known Bugs and Other Issues – In Loophole v0.2

Bug 01

Error in	NewLevel
Error	If shrinking the level, the level does not check if the player will be deleted on resize

Revision	December 7, 2009

Test Plan and Test Cases

Test Case	Case TC-1
Name	TC-1: Level Creation
Requirement	Level to run
Precondition	Level is loaded
Steps	Press button to enter edit mode Hit x to save level Reload level to verify changes
Exp Results	Level has changed to what was edited to
Revision	December 7, 2009; Changed button to save level

Test Case	Case TC-2
Name	TC-2: Player Movement
Requirement	Levels to run
Precondition	Level is loaded
Steps	Arrow keys to move
Exp Results	Character's position is different
Revision	

Test Case	Case TC-3
Name	TC-3: Level Succession
Requirement	Levels to run
Precondition	Level loaded
Steps	Place boxes on the platforms and hit space
Exp Results	Next level is loaded
Revision	December 7, 2009; Updated to reflect Sokoban implemented

Unit Testing

GameState

Function	How is it tested?
setNextState	states in the state machine are able to change
changeState	see above, requires setNextState to work
setPrevState	unimportant, proof of concept

GameInit

Function	How is it tested?
Constructor	Works successfully if Title shows up afterwards along with a display window

Title

Function	How is it tested?
Constructor	Title screen and images are displayed
doEvents	Keyboard events for space and Escape are possible
doRender	Screen does not have weird graphical errors if minimized due to frame buffer (meaning it is re-rendered)

Level

Function	
Function	How is it tested?
Constructor	Level map is loaded, no errors
doEvents	Keyboard input is accepted for a variety of events
doRender	screen updates properly
RenderBack	Back layer is rendered on screen
RenderFront	Front layer is rendered on screen
RenderObject	Player is rendered on screen
MovePlayer	Player moves and doesn't fall off of the array when moving
CreateBlock	Blocks can be created and take in possibilities of blank or existing blocks
Finish	Next level loads successfully
Resize	Level resizes and reloads with changes
ChangeGround	Ground texture beneath player changes
SaveLevel	Level can be saved and retrieved later
Pause	Not implemented (can be used for frame limiting, decided not to use)
ReturnIndex	Index is a decimal form of the hex value passed
nextLevel	The next level can load
ResetLevel	Level is restarted

Submitted Files

- images
  - background.jpg
  - logo.png
  - old-logo.png
- maps
  - map0.txt
  - map1.txt
  - map2.txt
  - map3.txt
  - map4.txt
- tiles
  - p.png
  - tiles.png
  - tiles.txt
- gameInit.cpp
- gameInit.h
- gameState.cpp
- gameState.h
- main.cpp
- Makefile
- NewLevel.cpp
- NewLevel.h
- structures.h
- Timer.cpp
- Timer.h
- Title.cpp
- Title.h

Software Demo

Loophole Engine with Sokoban Implemented
http://www.youtube.com/watch?v=7j4suLm3qjE

Final Codebase and Documentation Access

See attached file “Codebase.zip.” Located inside this file is the Readme file for Loophole

Executive Section

September 15, 2009

Untitled Project Zero (Loop Hole) `loop[0]`
Ryan Hagood
September 15, 2009
This week was primarily spent learning the SDL library and reading up on how to do various things related to input. After this week, I will include samples of the code I learned from throughout the week to teach me new things with notes on what I learnt and understood from each part. This will be useful for my notes as well as showing my progress in the learning aspect of my project.

September 22, 2009

Untitled Project Zero (Loop Hole)
Ryan Hagood
September 22, 2009
The past week was spent working on the data structures to be used to organize the information in the software package. These data structures primarily include level objects with tiles. Work also has been put into deciding on a map format which will be read in by the program for the creation of levels on the screen.

Time Spent

Data Structures: 6 hours
SDL: 3 hours
Total: 9 hours

Goals for Next Week

The goals for the next week are to create a simple movement system with collision detection onscreen between the player character and the environment. Another thing to work on is also coming up with a more interesting logo for the software package.

September 29, 2009

#loop[0]

Untitled Project Zero (Loop Hole)
Ryan Hagood
September 29, 2009
This past week was spent programming the collision detection in the maps so that the player can't go off of the map and crash the program along with the ability to push blocks around. Also this week a state machine was added for simple management of resources.

Time spent

Data Structures: 7 hours
SDL: 4 hours
Design: 3 hours
Total: 14 hours

Goals for Next Week

Allow loading of different levels and create different levels for the player to use. Also, learn more SDL Programming.

October 13, 2009

#loop[0]

Untitled Project Zero (Loop Hole)
Ryan Hagood
October 13, 2009
This past two week period was primarily spent working on learning more of the cross platform capabilities of SDL along with working with Subversion and Google Code. Not much progress was done at this time due to other tasks unrelated to the project.

Time spent

SDL: 5 hours
Subversion: 7 hours
Total: 12 hours
Total Overall: 49

Goals for Next Week

Allow loading of different levels and create different levels for the player to use. Also, learn more SDL Programming.

October 20, 2009

#loop[0]

Untitled Project Zero (Loop Hole)
Ryan Hagood
October 20, 2009
I have primarily spent this time working on emacs and organization

Time spent

Emacs: 2 hours
Subversion: 2 hours
Total: 4 hours
Total Overall: 53 hours

Goals for Next Week

November 3, 2009

Untitled Project Zero (Loop Hole)
Ryan Hagood
November 3, 2009

Description of Work

The current week was spent working on small items such as screen centering and level saving. The week was also spent thinking about the camera subsystem and the level resizing. In the coming week I hope to implement

Time Spent

Looking over code: 2 hours
Screen positioning: 1 hours
Level saving: 2 hours
Total: 5 hours
Total Overall: 62 hours

November 10, 2009

Ryan Hagood
November 10, 2009

Description of Work

This week I have been working to implement my camera system, however I have realized I have to reimplement my class for the level. Because of this, I am creating a new level class for this purpose. The new level class will replace the old one once it is working completely (it currently only displays the level on the screen). The reason for the camera system is to allow scrolling on the screen.

For next week I will further work to implement this new level class structure.

Time Spent

Trying to implement camera system: 2 hours
Restructuring Level.cpp (as NewLevel.cpp currently): 3 hours
Total Overall: 67 hours

November 17, 2009

Untitled Project Zero (Loop Hole)
Ryan Hagood
November 17, 2009

Description of Work

This week was spent trying to figure out surface to surface attachment (blitting). Finally after struggling for a great period of time I finally have figured this out (save for one or two problems). In the next two weeks I will be finishing up and fixing up all errors.

Time Spent

Working on tiling: 8 hours
Total: 8 hours
Total Overall: 70 hours

December 1, 2009

Ryan Hagood
December 1, 2009

Description of Work

The past two weeks I have spent finishing recoding most of the level subsystem. For the next week I will finish finalizing my code.

Restructuring Level.cpp (as NewLevel.cpp currently): 12 hours

Total Overall: 82 hours

December 8, 2009

Ryan Hagood
To: Project Director
December 8, 2009

Description of Work

This past week was spent working on commenting code and repairing bugs. During the previous week at the practice demo session people had mentioned they did not see the “game” in my project really, just messing around with blocks. This was resolved by implementing the rules for the game Sokoban which a block pushing game. This codebase and project summary should be enough to assist many people in making a basic puzzle game, with the framework provided. It has been nice working with you as my project director.

Time Spent

4 hours commenting
6 hours fixing bugs and implementing Sokoban
Total: 92 hours