https://en.wikipedia.org/wiki/Temple_Run

Rothberg ROASTERS

24783 Project

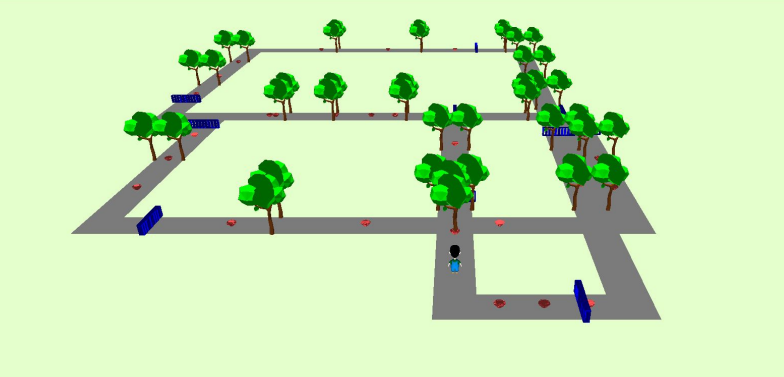
Rothberg Run!

Minjun Xu
Zi Li
Jing Zhao
Boxian Deng

Introduction

Rothberg RUN!

Github: https://github.com/rothberg-cmu/rothberg-run



Rothberg Run! is a keyboard-controlling temple-run-like game written in C++.

Keyboard Control:
[ENTER]: start/restart **[ESC]:** exit
[←]: shift left **[→]:** shift right **[SPACE]:** jump
[A]: turn left **[D]:** turn right

The **final score** is calculated based on the **mileage** and **diamonds** collected. Game ends when player is out of road and scores gets deducted everytime player encounters the **fence**. The player's history record will be stored in a .txt file and displayed in the **leaderboard** in the end.

Cross-platform Support

Linux

$ git clone https://github.com/rothberg-cmu/rothberg-run.git
$ cd rothberg-run
$ mkdir build
$ cd build
$ cmake ../src
$ cd 3d-construction
$ make
$ ./3d-construction

Windows

Download to local machine and go to the folder where you unzip the source code, make a folder called build which locates at the same level of /src folder and run following command in the Powershell.
$ cd build
$ cmake ../src
Then open Project.sln in your build folder with visual studio and compile the 3d-construction program.
Double click 3d-construction.exe and enjoy!

Mac

$ git clone https://github.com/rothberg-cmu/rothberg-run.git
$ cd rothberg-run
$ mkdir build
$ cd build
$ cmake ../src/ -G “Xcode”
$ open Project.xcodeproj
$ Debug/3d-construction

Player Introduction

Name: Rothberg, Cartoon boy, controlled by keyboard

Attributes:

position

orientation

velocity

Load binary

rotate

jump

scale

draw

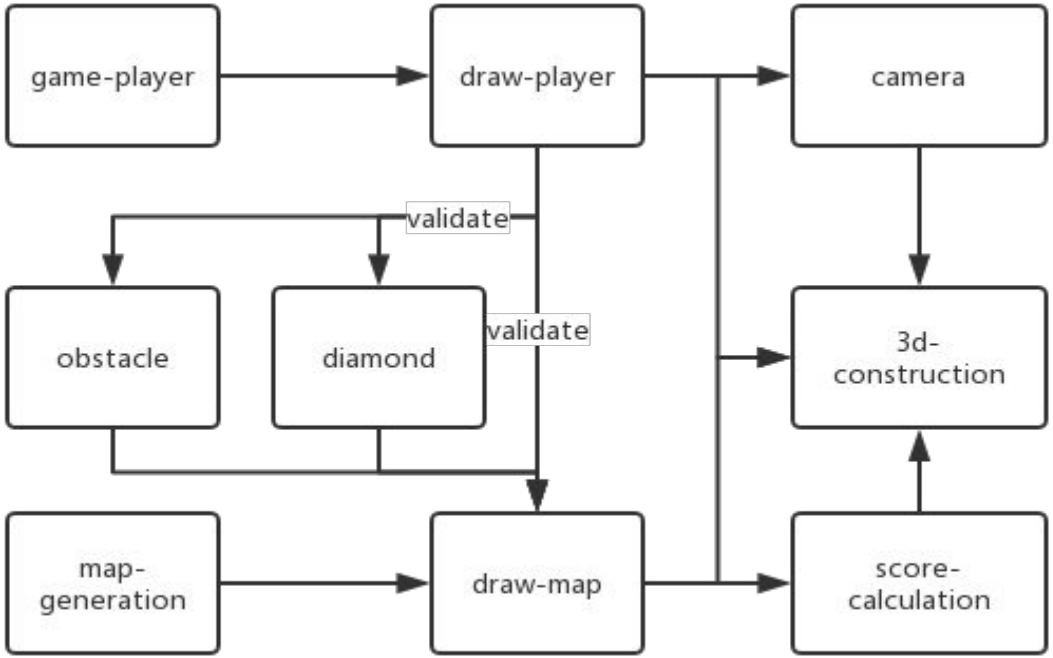
...

Jump mode

Alive status

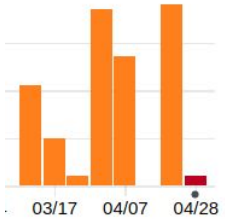
...

Pipeline


game-player
draw-player
camera
validate
obstacle
diamond
validate
3d-
construction
map-
generation
draw-map
score-
calculation

Contributions

260+ commits 20+ branches 4000+ lines of code excluding the external libraries



Commit history

Minjun Xu

- Map and roads generation
- Diamonds generation
- Obstacles generation

Zi Li

- Draw player
- Final integration of different components.

Jing Zhao

- Player generation
- Player motion design

Boxian Deng

- Draw roads and environments
- Scoreboard

Customer Survey

Customer Opinions

1. User should be able to jump over the obstacles.
2. Background can be image instead of blank.
3. There should be a scoreboard of some sort.
4. Add transition effect during turning left/right.
5. User should select difficulty level at the beginning.

Calibration

1. Added functionality of pressing space to jump.
2. Stored historical records and display in descending order.
3. Set palegreen background color.

Highlights

1. Developers' Side

1. Highly object-oriented
2. Highly modular and event-driven (based on Soji's framework)
3. Efficiently developing with pull-request and code review
4. Test Driven Development (TDD)
5. Easy to maintain and update

2. Customers' Side

1. Cross-platform support
2. Full access to game configuration (next version's feature)
3. User friendly with on-screen instructions

Trial Feedback

‘Beautiful environment, nice tree. Only perspective transition is too abrupt.’ --Nick

‘Too simple for me! Need different level of games’ --Justin

‘Nice character! If knock down happens after hitting the railing’ --Yan

Reference

3D models:

Cartoon Boy with hat by Mangesh Kondalkar, https://www.turbosquid.com/3d-models/boy-cartoon-hat-3d-dxf/1001332

Tree Cartoon Low Poly, created by Mark Alis, https://www.turbosquid.com/3d-models/3d-cartoon-tree/1045141

Diamond, created by BjarkeDuDe, https://www.turbosquid.com/FullPreview/Index.cfm/ID/675252

Fence, created by jenny-elephing, https://www.cgtrader.com/free-3d-models/interior/other/indoor-gate

Video

Q & A

Thanks for watching!