Laporan Mini Project

Game

“Block Dash”

By Muhammad Ariq Basyar

1806205110

Asdos: Giovan Isa Musthofa

Deskripsi Proyek:

Proyek ini adalah sebuah *game* yang diadaptasi dari geometry dash yaitu sebuah *game* di platform android dan ios. *Game* ini adalah permainan dua dimensi dengan kita sebagai sebuah *block* dan diharuskan untuk melompati semua *obstacle* yang bergerak ke arah kita, skor dihitung dari lamanya kita dapat bertahan dari seluruh *block* yang lewat. *Block dash* dapat menghitung dan menyimpan skor yang kita dapat dan khususnya skor terbaik (*best score*).

Deskripsi Library:

Salah satu *library* yang saya pakai adalah *library Pygame*, yaitu sebuah *library* yang memungkinkan kita membuat sebuah *game* dengan cara menumpuk semua label ke dalam *screen* dengan *mainloop* (*infinite while loop*). *Library* ini (*Pygame*) adalah *open source* untuk membuat aplikasi multimedia seperti permainan yang dibangun atas library SDL yang sangat baik. Seperti SDL, *pygame* sangat *portable* dan berjalan di hampir semua *platform* dan sistem operasi.

Selanjutnya, *library* yang saya pakai adalah *library* os yaitu sebuah *library* yang memungkinkan kita untuk mencari sebuah file di dalam *directory* dan kita dapat menemukan *absolute path*-nya, saya memakai *library* ini untuk mencari *file font* dan mencari *file log*, *about*, dan *file help*. *Library* ini sangat berguna untuk mencari *directory*, mencari file dengan mendapatkan absolute pathnya.

Selanjutnya library yang saya pakai adalah base64 yaitu sebuah *library* yang sangat berguna untuk meng-*encode* dalam RFC 3548. Standar ini mendefinisikan algoritma Base16,

Base32, dan Base64 untuk *encoding* dan *decoding string binary arbitrary* ke *string* teks yang dapat dengan aman dikirim melalui email, digunakan sebagai bagian dari URL, atau termasuk sebagai bagian dari permintaan HTTP POST. Saya memakai library ini untuk mengubah *best score* menjadi *encoded* agar tidak mudah diubah.

Selanjutnya *library* yang saya pakai adalah *sys* (*system*), *sys* adalah sebuah *library* python yang berfungsi untuk mengakses sistem. Modul ini menyediakan akses ke beberapa variabel yang digunakan atau dikelola oleh interpreter dan fungsi yang berinteraksi kuat dengan interpreter. Saya memakai *sys.exit()* untuk menghentikan *code* yang masih berjalan jika hanya memakai *pygame.quit()*.

Selanjutnya *library* yang saya pakai adalah *random*, *random* adalah sebuah modul yang dapat mengimplementasikan generator angka *pseudo-random* untuk berbagai distribusi matematika. *Random* ini sangat berfungsi untuk membuat *obstacle – obstacle* di dalam game saya yaitu dapat mengacak warna, letak x, letak y, dan banyak yang lainnya. *Library* ini sangat berguna untuk mengeluarkan sebuah angka yang sangat *random*.

Algoritma:

Pertama, yang akan saya bahas adalah bagaimana *mainloop* program ini bekerja. Saya membuat *mainloop* ini dengan cara terus menerus me-*replace* berbagai atribut ke *screen*, berbagai *text*, *button*, dan lain lain dengan *infinite while loop.* Saya juga membuat berbagai *class* seperti *class Game* (yaitu untuk membuka *window game* dan untuk merepresentasikan main program di dalam sana) yang saya simpan di dalam folder lain. Lalu ada *class Character*, *class* ini bertujuan untuk membuat *character* utama yaitu sebuah *block* yang dapat meloncat ke atas berbagai macam *obstacle*. Lalu ada *class Obstacle* yang bertujuan untuk membuat *obstacle* bergerak ke arah *user* dan membuatnya *random*.

Lalu, bagaimana saya mendapatkan data *high score*? Jawabannya adalah dengan cara menginisiasi *player score* sebagai 0 dan terus menambah nilai tersebut sampai *player* kalah atau *game over* dan saya simpan *highscore* ke *file* lain yaitu log.txt dan membuatnya ter-*encode* dengan baik menggunakan base64 *encoding*. Saya mempunyai beberapa *case* ketika *user* mengubah2 *file* log.txt, jika ada user lain yang mengubahnya tetap dalam base64, maka mungkin masih ada celah untuk bisa mengubah highscore. Tetapi jika yang dimasukkan adalah *string* yang tidak berarti, maka akan membuat high score menjadi tetap 0 (default), jika user

keluar dari *game*, maka akan otomatis menyimpan *high score* yang sedang dijalankan ke *file* log.txt.

Selanjutnya bagaimana saya membuat tombol berfungsi dengan baik? Pygame dapat membuat sebuah tombol dengan cara memasang *text* dan mengambil data *rectangle* yang dimiliki teks tersebut dan nantinya bisa digunakan untuk membuat sebuah tombol dari data x, y, lebar dan tinggi *rectangle* yang dimiliki oleh teks tersebut. Saya dapat mendeteksi *user* yang sedang klik sesuatu di dalam layar *game* dengan cara memanggil *list* yang berisi seluruh event *clicked* yang dilakukan oleh *user*/*player* di dalam *game* yaitu *built* in *function* *pygame*.*event*.*get*(), dan memanggil *boolean event*.*type* yang berisi tipe dari *event* yang dilakukan oleh *user*, jika *user* melakukan *click*, maka akan terekam tipe MOUSEBUTTONDOWN dari dalam *pygame* setelah saya mengetahui bahwa *user* sudah klik, saya harus cek bahwa posisi mouse berada di atas tombol dengan akurat yaitu dengan *event*.*pos* (mengambil posisi mouse) dan memakai *builtin boolean* *variabel_rectangle*.*collidepoint*(*mouse_pos*) yaitu mengecek apakah posisi mouse berada dalam *variabel_rectangle* (sebuah *variable* untuk merepresentasikan *rectangle* yang dimiliki tombol.

Selanjutnya saya akan menjelaskan bagaimana saya dapat membuat *character block* di *game* ini dapat meloncat, saya memakai sebuah *list* yang bernama “vel_naik” dan “vel_turun” berisi sebuah *list* yang berurut dari -40/3 sampai 0 dan 0 sampai 40/3 dengan masing – masing beda 0,667. Saya memiliki inisiasi nilai y di dalam *class Character* yang bertujuan untuk mengubah nilai y setiap saat ketika *player* sedang meloncat dan jika *list* “vel_naik” sudah habis, maka saya membuatnya kembali turun sampai menemukan *obstacle* lain atau menemukan *land*.

Lalu kapan permainan ini berakhir? saya membuat algo untuk selesainya *game* adalah ketika *block player* menabrak *block obstacle* dari bagian samping kiri dengan cara *boolean* jika nilai y *character* berada di antara nilai y *obstacle* dikurang 32 dan nilai y *obstacle* ditambah 35 maka akan *gameover* dan mengeluarkan *mainloop* (*infinite while loop*) *game* over dan mengeluarkan *player’s score* dan *best score*, jika *player’s score* lebih besar dari *best score*, maka *best score* akan otomatis berubah menjadi *player’s score*. Jika *player* menabrak bagian atas atau bawah, maka saya akan mengoptimalisasi nilai y pada *character* menjadi *dibagian* atas atau bawah *obstacle* (tidak *game* over). Saya juga membuat *game* semakin lama semakin cepat dengan menaikkan fps.