**DOKUMENTASI PROYEK AKHIR**

**PEMROGRAMAN GAME**

**PEMBUATAN GAME HAND-SANITY-ZER**

**[GAME LOGO HERE]**

**Dosen Pengampu**

**Eriq M.Adams J., S.T. M.Kom**

**Disusun Oleh Kelompok Semoga-A**

- 155150201111242 Gede Satria Harinamanata

- 155150200111079 Kurniawan setya pradana

- 155150207111196 Galang Izzatul Quddus

- 155150200111220 Arief Purba Sakti

- 155150200111303 Medeline Agustine

**FAKULTAS ILMU KOMPUTER**

**TEKNIK INFORMATIKA**

**UNIVERSITAS BRAWIJAYA**

**MALANG**

**2020**

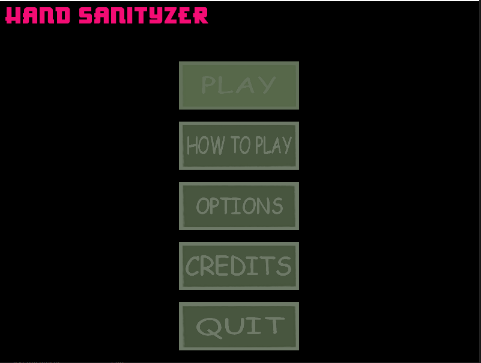
**INTRODUCTION**

Hand-SANITYzer adalah game yang bertemakan 1st person shooter. Game ini menceritakan sebuah waktu dimana pandemi virus sedang terjadi di berbagai belahan dunia. Game ini dibuat dengan tujuan untuk meningkatkan awareness terhadap pentingnya cuci tangan untuk mencegah tertularnya virus-COVID-20. Player dalam game ini diceritakan sebagai seorang yg mengidap “Mysophobia”, yang apabila virus gagal dibersihkan dari tangan, “Sanity” dari player akan berkurang dan player akan menjadi gila kemudian mati.

**GAME SETUP**

***Main Menu***

Tampilan Main menu akan langsung muncul ketika game dimulai. Main menu dapat di control dengan menggunakan pointer dan klik mouse untuk berinteraksi. Di dalamnya akan ada pilihan :

1. **Play :** Memulai game
2. **How To Play :** Menampilkan panduan bermain.
3. **Options :** Menanpilkan pengaturan-pengaturan yang dapat dilakukan dalam game
4. **Credits :** Menampilkan credits nama2 para creator game
5. **Quit :** Keluar dari game

***Options Menu***

Dalam Option player dapat merubah pengaturan dalam game, yakni sebagai berikut

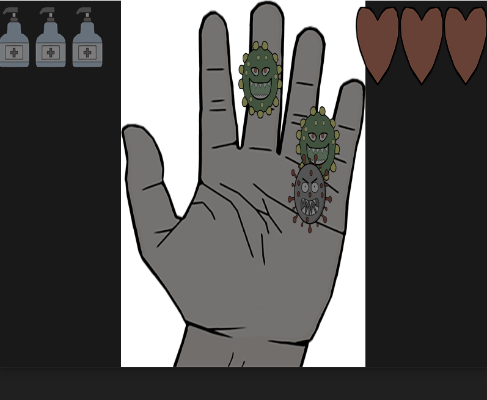
1. **Graphic :** Merubah pengaturan grafis dari game
2. **Audio :** Merubah pengaturan suara

***Credits Menu***

Menampilkan Credits dalam game. Dalam tampilan ini player hanya diberikan tombol “Back” untuk Kembali ke main menu

***How To Play Menu***

Menampilkan Panduan bermain kepada pemain.

****

***Gameplay Screen***

Gameplay akan menampilkan tangan dan virus yang akan terus bermunculan di tangan

***Result Screen***

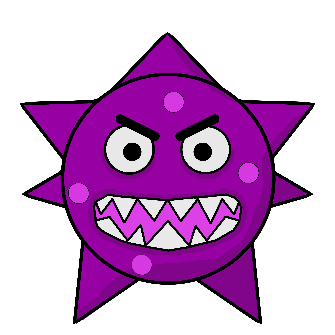
Menampilkan Hassil Akhir dari permainan.

**CHARACTER**

**The Hero**

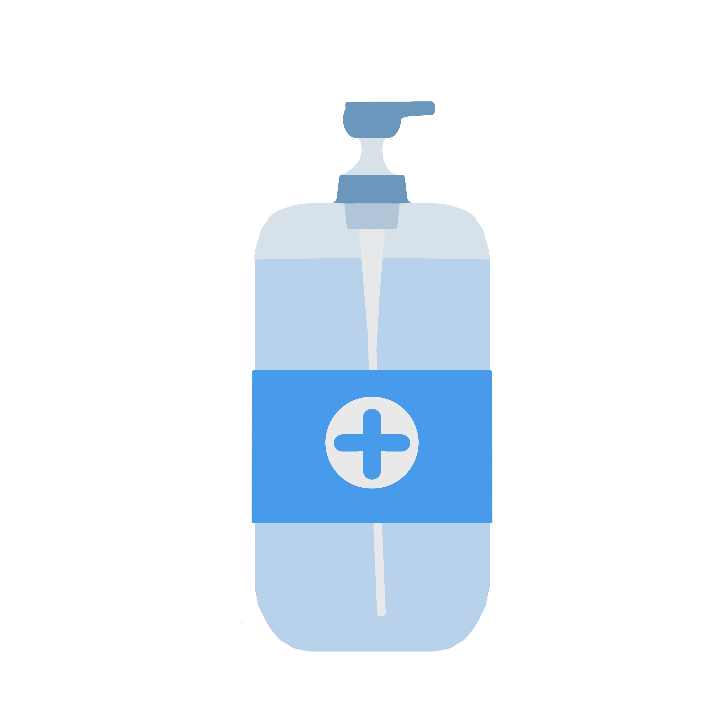
Ini dirimu, Kamu akan bermain sebagai seorang yang menderita “Mysophobia”. Kebersihan adalah motto mu. Kamu harus membersihkan tangan mu hingga steril.

**ENEMIES**

****

**COVID-20**

Virus ini akan menyerangmu, Senyum Jahat dan duri tajamnya siap untuk membunuhmu

****

**WEAPONS**

**101 SOAP**

Sabun DIY yang mampu melawan 101 jenis kuman, virus dan penyakit. Jumlah sabun sangat terbatas karena komposisi bahan-bahan telah terjual habis di w\*lmart

**SOURCE CODE**

A. Virus Game Object & Penampungan GameObject

|  |
| --- |
| VirusGameObject.h |
| #ifndef VIRUSGAMEOBJECT\_H  #define VIRUSGAMEOBJECT\_H  #include"GameObject.h"  class VirusGameObject :public GameObject  {  public:      VirusGameObject();      VirusGameObject(string SpriteName, string SpriteVertFrag, float x\_pos, float y\_pos, int screenWidth, int screenHeight, int Num\_Frames, float scale) : GameObject(SpriteName, SpriteVertFrag, x\_pos, y\_pos, screenWidth, screenHeight, Num\_Frames, scale) {};  //  ~VirusGameObject();      void Init();      void Update(float deltaTime);      void Render();      bool isAlive =false;      int alivetime=0;  };  #endif |

|  |
| --- |
| VirusGameObject.cpp |
| #include"VirusGameObject.h"  VirusGameObject::VirusGameObject() {  }  void VirusGameObject::Update(float deltaTime) {      this->GameObject::Update(deltaTime);      if (this->isAlive == false) {          this->ypos = -35;        }      else {          setPosx(getPosx());            setPosy(getPosy());          alivetime++;      }  }  void VirusGameObject::Render() {      this->GameObject::Render();  } |

VirusGameObject dibuat untuk menampung sprite Virus yang akan di tampilkan ke dalam game screen, dengan meng-include GameObject.h yang sudah ada dalam project yang berfungsi sebagai template object. Jadi VirusGameObject akan membuat method baru dengan menggunakan fungsi-fungsi dan parameter yang ada dalam GameObject.  
  
Penampungan object-object lainnya seperti Background, Hand, Health, dll. Dapat dilakukan dengan cara yang sama yaitu dengan melakukan pembuatan file .cpp dan .h baru yang nantinya hanya tinggal melakukan perubahan nama file dan method. Berikut adalah contoh dari BulletCountSprt yang menampung jumlah amunisi sabun dari player

|  |
| --- |
| BulletCountSprt.h |
| #ifndef BULLETCOUNTSPRT\_H  #define BULLETCOUNTSPRT\_H  #include "GameObject.h"  class BulletCountSprt: public GameObject  {  public:      BulletCountSprt();      BulletCountSprt(string SpriteName, string SpriteVertFrag, float x\_pos, float y\_pos, int screenWidth, int screenHeight, int Num\_Frames, float scale): GameObject(SpriteName, SpriteVertFrag, x\_pos, y\_pos, screenWidth, screenHeight, Num\_Frames, scale) {};      void Init();      void Update(float deltaTime);      void Render();      bool isAlive =false;  };  #endif |

|  |
| --- |
| BulletCountSprt.cpp |
| #include "BulletCountSprt.h"  BulletCountSprt::BulletCountSprt() {  }  void BulletCountSprt::Update(float deltaTime){      this->GameObject::Update(deltaTime);      if (this->isAlive == false) {          this->ypos = -35;      }      else {          setPosx(getPosx());          setPosy(getPosy());      }  }  void BulletCountSprt::Render() {      this->GameObject::Render();  } |

Dapat dilihat bahwa hanya perlu melakukan perubahan nama method saja dalam file baru untuk menampung object baru, pengerjaan ini menjadikan method untuk pemanggilan menjadi modular dan lebih mudah untuk memanggil object yg banyak apabila dalam game akan menggunakan object yang banyak.

B.GameplayScene

|  |
| --- |
| Gameplayscene.h |
| #ifndef GAMEPLAYSCENE\_H  #define GAMEPLAYSCENE\_H  #include "VirusGameObject.h"  #include "SceneHandler.h"  #include "Background.h"  #include "BulletCountSprt.h"  #include "HealthSprt.h"  enum gameState {      play = 0,      pause = 1,      gameOver = 2,      mainMenu = 3  };  class GameplayScene :public SceneHandler  {  public:      GameplayScene();      GameplayScene(unsigned int screenWidth, unsigned int screenHeight);  //  ~GameplayScene();      virtual void Init();      virtual void Update(float deltaTime);      virtual void Render();      //void updateGameOver(float deltaTime);      //void renderGameOver();  //  void updatePause(float deltaTime);  //  void renderPause();      void updatePlay(float deltaTime);      void renderPlay();      void IsClickedCheck(float x, float y);      void Reload();      gameState state = pause;      VirusGameObject VGO[7];      BulletCountSprt BCSprt[3];      Background bg;      HealthSprt HSprt[3];      int difficttimer=0;      int currentviruscount = 3;      int viruscount = 7;      int bulletCount = 3;      int playerHealth = 3;  };  #endif |

GameplayScene.h memanggil seluruh penampung dari object yang telah dibuat. Kemudian mendefinisikan State dari game, apakah game dalam kondisi play atau pause, dll. Kemudian terdapat penampung setting resolusi yang akan digunakan dalam GameplayScene.cpp.Di dalam kode juga berisikan konfigurasi-konfigurasi internal dari masing-masing GameObject,

|  |
| --- |
| GameplayScene.cpp |
| #include "GameplayScene.h"  GameplayScene::GameplayScene() {  }  GameplayScene::GameplayScene(unsigned int screenWidth, unsigned int screenHeight) {      this->screenWidth = screenWidth;      this->screenHeight = screenHeight;      Init();  }  void GameplayScene::Init() {        state = play;      for (int i = 0; i < viruscount; i++)      {          int temprollSprt = rand() % 3;          string tempsprtname;          switch (temprollSprt)          {          case 0:              tempsprtname = "virus1rez";              break;          case 1:              tempsprtname = "virus2rez";              break;          case 2:              tempsprtname = "virus3rez";              break;          default:              break;          }            VGO[i] = VirusGameObject(tempsprtname, "virusSprite", screenWidth/2 ,-30, screenWidth, screenHeight, 1, 1);          cout << "Sukses bool: " << VGO[i].isAlive << endl;        }      bg = Background("HandBG2", "virusSprite", 0, 0, screenWidth, screenHeight, 1, 1);      bg.SetPosition(((float)screenWidth - bg.frame\_width) / 2, ((float)screenHeight - bg.frame\_height) / 2);      for (size\_t i = 0; i < bulletCount; i++)      {          BCSprt[i] = BulletCountSprt("Bullet", "virusSprite", screenWidth / 2, -30, screenWidth, screenHeight, 1, 1);          if (i!=0)          {              BCSprt[i].SetPosition(BCSprt[i - 1].xpos + BCSprt[i - 1].frame\_width+10, 10);          }else          BCSprt[i].SetPosition(0, 10);      }      for (size\_t i = 0; i < playerHealth; i++)      {          HSprt[i] = HealthSprt("HealthFix", "virusSprite", screenWidth / 2, -30, screenWidth, screenHeight, 1, 1);          if (i != 0)          {              HSprt[i].SetPosition(HSprt[i - 1].xpos - HSprt[i - 1].frame\_width, 10);          }          else              HSprt[i].SetPosition(screenWidth-HSprt[i].frame\_width, 10);      }  }  void GameplayScene::Render() {      switch (state) {      case play:          renderPlay();          break;      case pause:          break;      case gameOver:          break;      default:          break;      }  }  void GameplayScene::Update(float deltaTime) {      switch (state)      {      case play:          updatePlay(deltaTime);          break;      case pause:          //updatePause(deltaTime);          break;      case gameOver:          //updateGameOver(deltaTime);          break;      default:          break;      }  }  void GameplayScene::updatePlay(float deltaTime) {      for (int i = 0; i < currentviruscount; i++)      {          if (VGO[i].isAlive == false)          {              int tempPostx = rand() % (int)bg.xpos + (bg.frame\_width - VGO[i].frame\_width);              int tempPosty = rand() % (int)(bg.frame\_height - VGO[i].frame\_height);              VGO[i].xpos = tempPostx;              VGO[i].ypos = tempPosty;              VGO[i].isAlive = true;              cout << "isalive: " << VGO[i].isAlive << endl;          }          else {              if (VGO[i].alivetime > 500)              {                  cout << "HITPLAYER" << endl;                  VGO[i].isAlive = false;                  VGO[i].alivetime = 0;                  if (playerHealth > 1)                  {                      playerHealth -= 1;                  }                  else                  {                      cout << "game over" << endl;                  }              }          }          //int tempPostx = rand() % (((screenWidth / 2 + 120) + 1) - (screenWidth / 2 - 220)) + (screenWidth / 2 - 220);          VGO[i].Update(deltaTime);      }      difficttimer++;      if (difficttimer > 1000)      {          if (currentviruscount < viruscount)          {              currentviruscount++;          }          difficttimer = 0;      }  }  void GameplayScene::renderPlay() {      bg.Render();      for (size\_t i = 0; i < playerHealth; i++)      {          HSprt[i].Render();      }      for (size\_t i = 0; i < bulletCount; i++)      {          BCSprt[i].Render();      }      for (int i = 0; i < currentviruscount; i++)      {          VGO[i].Render();      }  }  void GameplayScene::IsClickedCheck(float x, float y) {      for (size\_t i = 0; i < viruscount; i++)      {          if (VGO[i].IsClicked(x,y))          {                  cout << "VGO number " << i << " clicked" << endl;                  VGO[i].isAlive = false;                  VGO[i].alivetime = 0;            }      }  }  void GameplayScene::Reload() {        bulletCount = 3;      cout << "reloaded "<< bulletCount << endl;  } |

Dalam GameplayScene.cpp terdapat setting dari resolusi, setup dari setiap object yang akan dimasukan ke dalam game mulai dari nama sprite gambar, posisi x dan y gambar, lebar dan tinggi gambar hingga jumlah frame berserta scaling ukuran dari sprite.   
  
Dalam GameplayScene.cpp juga berisikan behavior-behavior dari setiap object yang masuk ke dalam scene. Seperti apa yang terjadi apabila object berinteraksi dengan klik sehingga terjadi perubahan frame pada object tersebut.