UNIVERSITAS

# **Modul 1 - GameCapstone**

# Desain Komponen Permainan



### Didukung oleh:



## 1 DESAIN KOMPONEN PERMAINAN

Pengembangan video game bukanlah suatu hal yang mudah dan sederhana karena biasanya memiliki beberapa langkah-langkah, dimulai dari pencarian ide, pembuatan game design document, hingga pemrograman game tersebut. Terdapat banyak komponen untuk membuat suatu game, beberapa komponen-komponen yang penting dalam pembuatan game antara lain: game flow, game states, game loops, game object, game character, game animation, dan game Al. Setiap komponen ini saling bersinergi satu sama lain untuk menciptakan game yang menarik dan menyenangkan untuk dimainkan.

Pada modul ini, akan diberikan contoh pengembangan *video game* sederhana dengan menggunakan perangkat lunak *Visual Studio 2015*.

#### **Tujuan Modul**

- 1. Mampu mengembangkan game flow, game state, dan game loops.
- 2. Mampu membuat objek dan karakter dalam permainan.
- 3. Mampu menghasilkan animasi dan kecerdasan buatan dalam permainan.

#### 1.1 Ringkasan Pembelajaran

Video game merupakan kumpulan dari beberapa komponen game yang saling mendukung satu sama lain sehingga menghasilkan game yang menarik untuk dimainkan. Namun, sebelum pengembangan video game dilakukan, dibutuhkan suatu dokumen mengenai game design dari video game yang akan dikembangkan. Beberapa hal dasar yang dilalui untuk membuat desain suatu video game antara lain (LaMothe, 2017):

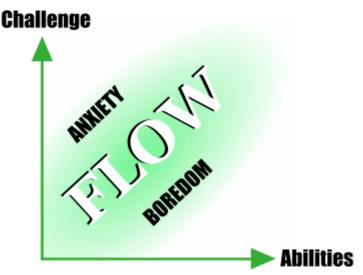
- Mencari ide.
- Membuat storyboard dan rough sketches dari dunia video game, karakter, level, dan lainnya.
- Mendaftar seluruh detail dari video game.
- Menghasilkan game design document.

Dalam game design document akan terlihat dengan lebih jelas beberapa komponen penting yang harus terdapat dalam video game yang akan dikembangkan seperti:

- Game flow, game state, dan game loops.
- Game object dan game character.
- Game animation dan game AI (Artificial Intelligence).

#### 1.1.1 Game Flow, Game State, dan Game Loops

Dapat dipastikan ketika pemain yang memiliki kemampuan yang tinggi diberikan tantangan yang mudah, pemain tersebut akan cepat merasa bosan. Sebaliknya, ketika pemain memiliki kemampuan yang rendah diberikan tantangan yang sulit, pemain tersebut akan cepat merasa frustasi, tetapi ketika pemain tersebut diberikan tantangan yang sesuai dengan kemampuan pemain saat itu, maka permainan tersebut dapat dikatakan permainan yang bagus, dikarenakan permainan tersebut akan memberikan tingkat *fun* yang tinggi. (Schreiber, 2009)



Gambar 1. 1 Gambaran Grafik Game Flow (Koven, 2017)

Menjaga tingkat keseimbangan antara kemampuan dan tantangan dalam permainan merupakan kunci untuk meningkatkan pengalaman *fun* bagi pemain. Tetapi hal ini kadang cukup sulit dilakukan dikarenakan setiap pemain memiliki kemampuan yang berbeda-beda seperti seseorang dapat memiliki kemampuan bermain catur lebih baik dibandingkan dengan orang lain. Selain itu, setiap pemain dapat mengalami peningkatan dalam kemampuan yang mereka miliki seperti seseorang yang terus-menerus berlatih dalam satu bidang dapat mengalahkan orang lain yang sebelumnya lebih kuat dari dia. Maka dari itu, dibutuhkan tingkat keseimbangan yang dinamis, yaitu tingkat kesulitan yang cukup kompleks tetapi membutuhkan tantangan yang dapat diatasi. (Koven, 2017)

Langkah lain yang dapat diterapkan adalah diberikannya peningkatan kesulitan permainan. Semakin tinggi kemampuan pemain, pemain tersebut akan diberikan akses menuju tingkat kesulitan permainan yang lebih tinggi atau dengan memberikan pililhan tingkat kesulitan yang berbeda ketika memulai permainan. Pemberian tingkat kesulitan yang berbeda-beda atau meningkat secara bertahap memberikan pemain kemampuan untuk mempelajari peraturan dalam permainan dan juga waktu yang dibutuhkan untuk menguasai kemampuan tersebut.

Memiliki permainan yang memberikan tingkat kesulitan sesuai dengan kemampuan pemain berarti permainan tersebut memiliki *game flow* yang baik. Dengan adanya *game flow* yang baik, maka permainan tersebut juga akan memberikan tingkat *fun* yang tinggi. (Koven, 2017)

Untuk menjaga *game flow* yang baik, membutuhkan penempatan *game state* yang berupa setiap objek dalam permainan dengan jelas dan benar. Objek yang dimaksud ialah ketika membuka *menu*, ketika permainan di-*pause*, ketika karakter bergerak dan menembak, ketika mengalahkan musuh, ketika mendapatkan *item*, dan sebagainya.

Game state biasanya memiliki tingkat pemrograman yang mudah atau sulit. Hal ini dikarenakan adanya perbedaan variabel yang dibutuhkan dalam setiap game state yang berbeda. Ketika melakukan pause, variabel yang dibutuhkan mungkin hanya berupa satu boolean saja, tetapi ketika membuka menu, variabel yang dibutuhkan dapat berupa beberapa boolean, array, string, dan integer. Dengan penempatan game state yang sesuai dengan objek dan kebutuhannya, maka permainan juga akan semakin terlihat masuk akal dan menarik. (Clingerman, 2008)

Selain *game state* dan *game flow*, terdapat satu komponen lagi yang membuat *game* dapat berjalan, yaitu *game loop*. Semua pemrograman dalam permainan berjalan di dalam *game loop*, yang berjalan secara terus-menerus selama permainan tersebut dimainkan. (PotHix, 2012) Terdapat tiga fase yang terjadi dalam *game loop*, yaitu:

#### 1. Initialize

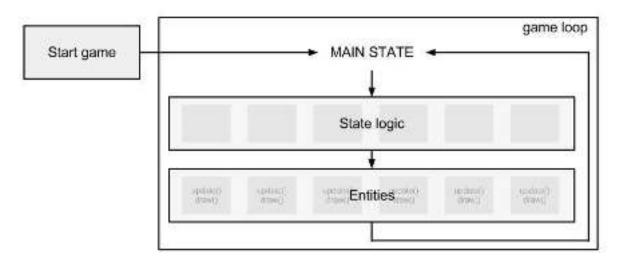
Fase ini mempersiapkan seluruh *game setup* yang penting dan *environment* untuk fase *update* dan *draw*. Tujuan utama dalam fase ini ialah untuk membuat entitas utama, mempersiapkan *menu*, mendeteksi komponen *hardware*, dan fungsi lainnya.

#### 2. Update

Fase ini bertujuan untuk mempersiapkan semua objek untuk di-*draw*. Pada fase ini dijalankan *code* fisik, *update* koordinat, peningkatan karakter, kerusakan, dan operasi lainnya. Pada fase ini *input* pemain diterima dan diproses.

#### 3. Draw

Fase ini dijalankan untuk men-*draw* setiap informasi yang sudah dipersiapkan sebelumnya ke dalam layar. Fungsi yang dijalankan seharusnya mengandung keseluruhan pemrograman untuk mengatur dan men-*draw level, layer,* karakter, *HUD,* dan objek lainnya.



Gambar 1. 2 Gambaran siklus game loops (PotHix, 2012)

Ketika fase *draw* dijalankan, berarti seluruh entitas (karakter, objek, dan lainnya) dalam permainan akan memperbarui *properties* (status, level, posisi, dan lainnya) mereka dan meproyeksikannya pada permainan.

#### 1.1.2 Object dan Character dalam Permainan

Object dalam permainan merupakan semua objek yang dapat dilihat atau diinteraksi dalam permainan, misalnya tembok, senjata, musuh, *power up*, dan lainnya. Setiap *object* memiliki kemampuan dan fungsi yang berbeda-beda bergantung dari pemrograman yang diberikan pada *object* tersebut. (Game Objects, 2017)

Sebuah *object* dapat berfungsi sebagai tempat berpijak atau dengan pemrograman yang lebih kompleks dapat berupa tempat untuk melakukan penyimpanan data, tempat untuk berpindah lokasi, atau lainnya. (Gamefroot, 2017)



Gambar 1. 3 Screen Shot Game Age of Empire (Microsoft, Age of Empires, 2017)

Pada *game* Age of Empires, terdapat berbagai macam *object* dengan fungsi yang berbedabeda. Pepohonan diprogram untuk memberikan sumber daya kayu, sungai diprogram sebagai tempat yang dikhususkan untuk kapal, dan masih banyak *object* lainnya yang memiliki fungsi tertentu.

Object dalam permainan yang diprogram lebih kompleks dibandingkan object lainnya disebut sebagai character. Media game, dibandingkan dengan media lain, sangat bergantung pada identifikasi dari karakter dalam permainan. Seperti ketika terdapat adaptasi novel menjadi film, fans akan merasa sedikit aneh melihat fitur yang terdapat dalam kedua media berbeda. Hal seperti ini yang menjadi alasan game membuat pemainnya untuk mendesain karakter mereka sendiri. Karena, jika karakter yang dimainkan sesuai dengan yang didesain, maka pemain juga akan merasa lebih nyaman ketika bermain. (Alderman, 2015)



Gambar 1. 4 Karakter Guybrush Threepwood dari seri Monkey Island (Alderman, 2015)

Character tidak hanya terbatas pada character yang dapat dikendalikan oleh pemain, tetapi terdapat juga beberapa jenis character lainnya dalam permainan, antara lain:

- Player character

Karakter dalam permainan yang diprogram untuk dapat dikendalikan oleh pemain, biasanya berupa karakter utama dalam permainan.

- Alternate character

Karakter yang dapat dikendalikan oleh pemain, disamping dari karakter utama yang dikendalikan.

Non-player character (NPC)

Karakter yang tidak dapat dikendalikan oleh pemain, *NPC* dapat berupa karakter yang memberikan informasi, menjual *item*, memberikan *quest*, atau lainnya.

- Boss character

Karakter yang tidak dapat dikendalikan oleh pemain, berupa bos musuh dalam permainan, biasanya karakter bos memiliki program yang cukup kompleks dibandingkan dengan *mob character*.

Mob character

Karakter yang tidak dapat dikendalikan oleh pemain, berupa monster yang memiliki pemrograman yang tidak terlalu kompleks.

#### 1.1.3 Animasi dan Kecerdasan Buatan dalam Permainan

Media *game* dimaksudkan untuk interaktif, yaitu ketika dimainkan, pemain memiliki kendali penuh akan karakter yang dimainkan termasuk kamera. Pemain yang menjalankan cerita dan menggerakan karakter. Sehingga tidak hanya program harus berjalan dengan benar, tetapi tampilannya juga harus terlihat halus (Pluralsight, 2014).



Gambar 1. 5 Animasi 2D dari game In Between (Daniel, 2015)

Animasi dapat dipisahkan paling tidak menjadi dua kategori (Mahinder, 2016):

#### 1. Animasi 2D

Animasi 2D merupakan bentuk animasi yang sudah ada sejak lama dan animasi ini tidak mencoba untuk mengimitasi kehidupan nyata.

Beberapa kelebihan dari penggunaan animasi 2D antara lain:

- o Biaya produksi yang lebih rendah
- Lebih cepat untuk diproduksi
- o Lebih mudah dan sederhana
- Mudah untuk dikendalikan
- o Lebih berfokus pada *game play*

Beberapa kekurangan dari penggunaan animasi 2D antara lain:

- o Lebih cepat merasa bosan
- o Permintaan yang lebih sedikit
- Membutuhkan waktu yang lama untuk membuat template animasi yang dapat digunakan secara berulang

#### 2. Animasi 3D

Animasi 3D memberikan rasa nyata bagi pemainnya, pemain seperti merasakan apa yang terjadi sebenarnya pada karakter 3D sehingga beberapa pemain merasa lebih menyukai animasi 3D terutama untuk *game action*.

Beberapa kelebihan dari penggunaan animasi 3D antara lain:

- o Memiliki pergerakan yang lebih realistis
- o Animasi 3D dapat digunakan secara berulang
- o Memberikan potensi pergerakan yang lebih banyak
- o Memiliki lebih banyak permintaan

Beberapa kekurangan dari penggunaan animasi 3D antara lain:

- o Imajinasi dalam pembuatan animasi terbatas
- o Tingkat kekompleksan yang tinggi

Setiap animasi, baik 2D atau 3D memiliki kelebihan dan kekurangan masing-masing sehingga pengembang harus menentukan animasi apa yang akan digunakan bergantung kebutuhan dari permainan yang akan dikembangkan.

Animasi menjadi kunci untuk membuat *game* yang menarik dengan mengutilisasi *concept* design, character design, motion graphics, realistic facial expressions, dan pewarnaan. Animasi dalam *game* membuat adanya peningkatan dalam komplekitas, keindahan, dan popularitas. (PowerHouse Animation Studio, 2015)

Selain animasi, g*ame* juga tidak terlepas dari dibutuhkannya kecerdasan buatan. Kecerdasan buatan yang dimaksud dapat terlihat dengan adanya NPC dan monster dalam permainan. Dimulai dari satu fungsi sederhana hingga beberapa fungsi yang kompleks. (Eastwood, 2017)

Kecerdasan buatan yang diberikan pada NPC biasanya terpusat pada tingkat kepintaran dan peningkatan *game play* dengan *batasan environment*. Kecerdasan buatan yang diberikan berbeda dengan yang diberikan pada karakter bahkan penggunaan *cheat* juga diperbolehkan, misalnya pada permainan strategi yang membutuhkan pembangunan markas, NPC diberikan *cheat* untuk dapat melihat keseluruhan peta dan terkadang kemampuan NPC harus dikurangi sehingga tidak membebani pemain. Sebagai contoh, pada permainan *first-person shooter*, NPC tidak diberikan kemampuan untuk melakukan *head-shot* yang merupakan kemampuan yang melebihi kemampuan pemain pada umumnya. (Seemann & Bourg, 2004)

#### GameCapstone

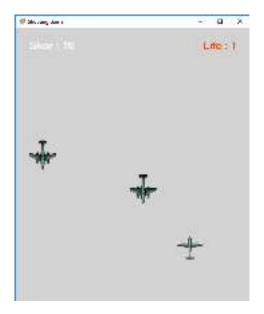


Gambar 1. 6 Screen Shot game Gears of Wars (Microsoft, Gears of War: Ultimate Edition , 2016)

Secara umum, hampir semua *game* menggunakan kecerdasan buatan. Bahkan untuk objekobjek yang tidak terlihat hidup, seperti hantu pada permainan Pac Man hingga *bots* pada permainan *first-person shooter*.

#### 1.2 Video Tutorial URL

Game yang akan dikembangkan pada modul ini berupa Game Shooter sederhana menggunakan Visual Studio 2015.

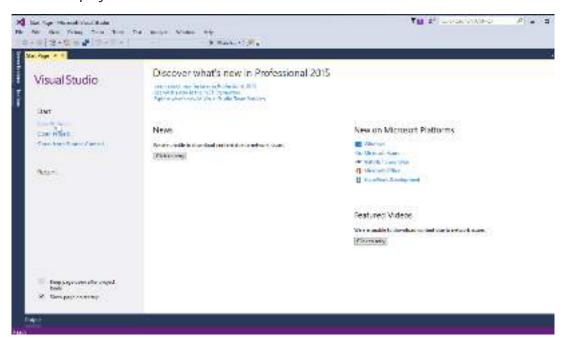


Gambar 1. 7 Tampilan Shooting Game

Tautan Video Tutorial: <a href="https://youtu.be/C6C1W37dEXM">https://youtu.be/C6C1W37dEXM</a>.

#### 1.2.1 Langkah 1-Membuat New Project

Ketika menjalankan Visual Studio 2015, akan terbuka halaman *welcome* sebelum membuka atau membuat *project* baru.

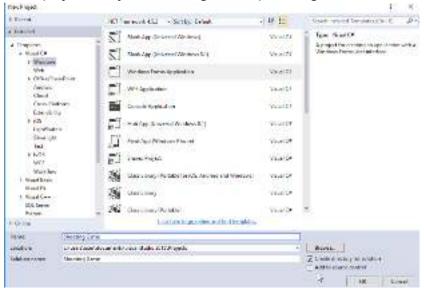


Gambar 1. 8 Halaman Awal Visual Studio 2015

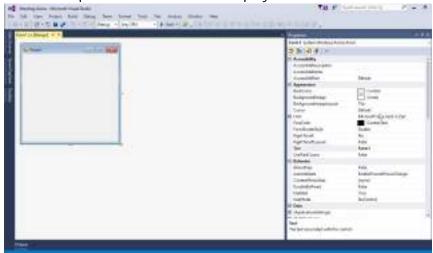
1. Klik menu "New Project..."



2. Kemudian pilih, 'Template -> Visual C# -> Windows Forms Application'. Juga ubah nama *project* menjadi 'Shooting Game' pada bagian "Name". Lalu klik tombol "OK"



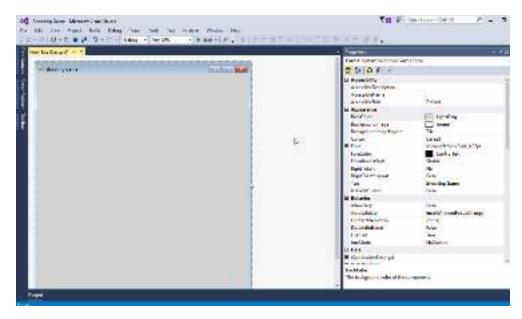
3. Berikut tampilan Visual Studio ketika project baru dibuat.



#### 1.2.2 Langkah 2-Mempersiapkan Komponen-Komponen yang Dibutuhkan

Terdapat beberapa komponen yang harus dipersiapkan dalam pembuatan 'Shooting Game'

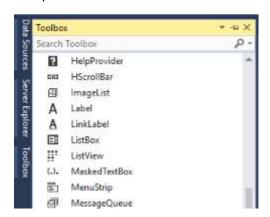
#### Jendela Permainan



Jendela permainan merupakan tampilan kotak 'Form1'. Untuk mengubah tampilan 'Form1' lakukan pengaturan 'properties' seperti berikut:

- 1. Ubah bagian 'Layout -> Size' menjadi '540, 640'.
- 2. Ubah bagian 'Appearance -> Text' menjadi 'Shooting Game'.
- 3. Ubah bagian 'Appearance -> BackColor' menjadi 'Web -> LightGray'.

#### Komponen Permainan



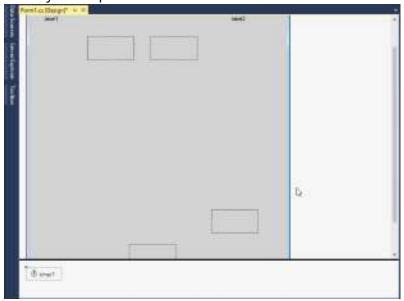
Komponen permainan merupakan komponen-komponen yang akan digunakan untuk membuat permainan dapat dijalankan, komponen-komponen ini dapat diambil dari bagian jendela 'Toolbox'

1. Persiapkan komponen-komponen yang akan digunakan.

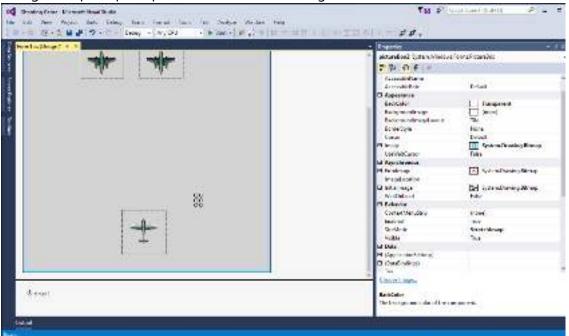


c. 1 Timer

Susun komponen-komponen tersebut dengan menarik setiap komponen dari 'Toolbox' ke dalam jendela permainan



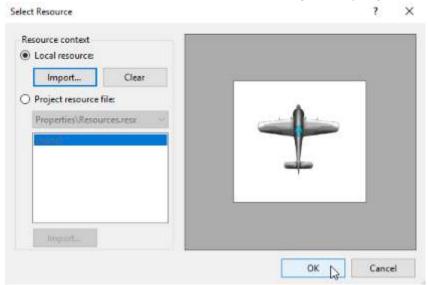
2. Ubah pengaturan pada setiap 'PictureBox' yang sudah dipersiapkan. PictureBox berfungsi sebagai tampilan pada permainan dalam bentuk gambar.



- a. Player
  - i. Ubah gambar 'PictureBox'. Klik tanda segitiga untuk membuka tampilan berikut

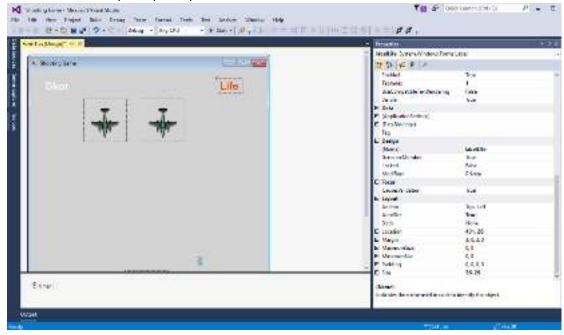


Kemudian pilih 'Local resource -> Import...' untuk membuka gambar dari folder di dalam komputer. Jika sudah memilih gambar yang sesuai, klik 'OK'.



- ii. Ubah pengaturan 'properties' berikut:
  - 1. "Behavior -> Size Mode" menjadi "StretchImage".
  - 2. "Layout -> Size" menjadi "100,100".
  - 3. "Design -> Name" menjadi "player".
- b. Enemy (2 PictureBox)
  - i. Ubah gambar 'PictureBox'.
  - ii. Ubah pengaturan 'properties' berikut:
    - 1. "Behavior -> Size Mode" menjadi "StretchImage".
    - 2. "Layout -> Size" menjadi "100,100".
    - 3. "Design -> Name" menjadi "enemy01" dan "enemy02".
- c. Bullet
  - i. Ubah gambar 'PictureBox'.
  - ii. Ubah pengaturan 'properties' berikut:
    - 1. "Behavior -> Size Mode" menjadi "StretchImage".
    - 2. "Layout -> Size" menjadi "9,19".
    - 3. "Design -> Name" menjadi "bullet".

3. Ubah pengaturan pada setiap 'Label' yang sudah dipersiapkan. Label berfungsi untuk memberikan tampilan pada permainan dalam bentuk tulisan.



- a. Label Skor
  - i. Ubah pengaturan 'properties' berikut:
    - 1. "Appearance -> Text" menjadi "skor".
    - 2. "Appearance -> ForeColor" menjadi "Web -> White".
    - 3. "Appearance -> Font -> FontSize" menjadi "18".
    - 4. "Appearance -> Font -> Bold" menjadi "true".
    - 5. "Design -> Name" menjadi "labelSkor".
- b. Label Life
  - i. Ubah pengaturan 'properties' berikut:
    - 1. "Appearance -> Text" menjadi "life".
    - 2. "Appearance -> ForeColor" menjadi "Web -> OrangeRed".
    - 3. "Appearance -> Font -> FontSize" menjadi "17".
    - 4. "Appearance -> Font -> Bold" menjadi "true".
    - 5. "Design -> Name" menjadi "labelLife".
- 4. Ubah pengaturan pada 'Timer' yang sudah dipersiapkan. Timer berfungsi sebagai pewaktu dalam permainan ('Timer' berperan sebagai 'FPS').



- a. Ubah pengaturan 'properties' pada timer, menjadi sebagai berikut:
  - i. "Behavior -> enabled" menjadi "true".
  - ii. "Behavior -> Interval" menjadi "90".

iii. "Design -> Name" menjadi "timerPermainan".

#### 1.2.3 Langkah 3-Pemrograman Inisialisasi Variabel Awal Permainan

Buka jendela pemrograman/code dengan mengklik kanan pada jendela design kemudian memilih "View Code" atau dapat juga dengan menggunakan tombol "F7".

1. Tambahkan kode berikut untuk mempersiapkan variabel yang akan digunakan dalam pemrograman.

```
public partial class Form1 : Form
{
    int pergerakanKiri = 0;
    int kecepatanPeluru = 8;
    bool dapatMenembak = true;
    int skorPermainan = 0;
    int nyawa = 3;

    int pergerakanEnemy = 5;
    Random angkaAcak = new Random();

    System.Media.SoundPlayer efekTembakan = new System.Media.SoundPlayer();
}
```

2. Tambahkan kode berikut untuk menginisialisasi beberapa variabel awal permainan

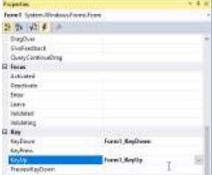
```
public Form1()
{
    InitializeComponent();

    enemy01.Top = -500;
    enemy02.Top = -1200;
    bullet.Top = -100;
    bullet.Left = -100;

    efekTembakan.SoundLocation = @"C:\Users\USER\Pictures\resource\shot.wav";
    efekTembakan.Load();
}
```

#### 1.2.4 Langkah 4-Pemrograman Input Permainan

Pastikan pembuatan *input key* dimulai dari pembuatan fungsi kosong yang diberikan oleh visual studio secara *default*.



1. Kembali ke tampilan jendela *design* kemudian pada bagian 'properties' klik tombol 'events' (tombol yang berbentuk halilintar).

- 2. Lalu cari bagian 'Key -> KeyDown' dan 'Key -> KeyUp'.
- 3. Pada bagian kosong di sebelahnya, klik tombol 'Enter'.

Setelah fungsi kosong sudah dibuat, tambahkan kode berikut untuk memrogram *input* permainan.

1. Tambahkan kode berikut untuk menerima *input* bahwa pemain menekan tombol, kode berikut akan membuat karakter bergerak ketika menerima *input* arah kiri atau arah kanan, dan menembak ketika menerima *input* spasi.

```
private void Form1 KeyDown(object sender, KeyEventArgs e)
    if(e.KeyCode == Keys.Left)
        if (player.Location.X < 0)</pre>
            pergerakanKiri = 0;
        }
        else
            pergerakanKiri = -5;
    else if (e.KeyCode == Keys.Right)
        if (player.Location.X > 530)
            pergerakanKiri = 0;
        else
            pergerakanKiri = 5;
    else if (e.KeyCode == Keys.Space)
        if(dapatMenembak == true)
        {
            kecepatanPeluru = 8;
            bullet.Left = player.Left + 50;
            bullet.Top = player.Top;
            dapatMenembak = false;
            efekTembakan.Play();
        }
    }
```

2. Tambahkan kode berikut untuk menerima *input* bahwa pemain melepas tombol. Kode berikut akan membuat karakter bergerak ketika *input* arah kanan atau kiri dihentikan

```
private void Form1_KeyUp(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Left)
    {
        pergerakanKiri = 0;
    }
    else if (e.KeyCode == Keys.Right)
    {
        pergerakanKiri = 0;
    }
}
```

#### 1.2.5 Langkah 5-Pemrograman Fitur Permainan

Fitur permainan diberikan melalui fungsi kosong yang dihasilkan dari meng-klik 2 kali 'timer' yang terdapat pada bagian jendela *design*.



Terdapat beberapa fitur yang harus diberikan.

1. Kode berikut merupakan bagian kode inti, yang melakukan *looping*, yaitu dijalankan secara terus-menerus.

```
private void timerPermainan_Tick(object sender, EventArgs e)
{
    player.Left += pergerakanKiri;
    bullet.Top -= kecepatanPeluru;
    enemy01.Top += pergerakanEnemy;
    enemy02.Top += pergerakanEnemy;
    labelSkor.Text = "Skor : " + skorPermainan;
    labelLife.Text = "Life : " + nyawa;

    if (!dapatMenembak && bullet.Top <0)
    {
        dapatMenembak = true;
        kecepatanPeluru = 0;
        bullet.Top = -100;
        bullet.Left = -100;
    }

    enemyHit();
    playerHit();
}</pre>
```

2. Kode berikut berfungsi untuk mengecek apakah 'bullet' bersentuhan dengan 'enemy'.

```
private void enemyHit()
{
    if (bullet.Bounds.IntersectsWith(enemy01.Bounds))
    {
        skorPermainan += 1;
        enemy01.Top = -500;
        int posisiAcak = angkaAcak.Next(1, 400);
        enemy01.Left = posisiAcak;
        dapatMenembak = true;
        kecepatanPeluru = 0;
        bullet.Top = -100;
        bullet.Left = -100;
   else if (bullet.Bounds.IntersectsWith(enemy02.Bounds))
        skorPermainan += 1;
        enemy02.Top = -1200;
        int posisiAcak = angkaAcak.Next(1, 400);
        enemy02.Left = posisiAcak;
        dapatMenembak = true;
        kecepatanPeluru = 0;
        bullet.Top = -100;
        bullet.Left = -100;
    }
```

3. Kode berikut berfungsi untuk mengecek apakah 'player' bersentuhan dengan 'enemy'.

```
private void playerHit()
    if (enemy01.Bounds.IntersectsWith(player.Bounds))
    {
        nyawa -= 1;
        enemy01.Top = -500;
        int posisiAcak = angkaAcak.Next(1, 400);
        enemy01.Left = posisiAcak;
    else if (enemy02.Bounds.IntersectsWith(player.Bounds))
        nyawa -= 1;
        enemy02.Top = -1200;
        int posisiAcak = angkaAcak.Next(1, 400);
        enemy02.Left = posisiAcak;
    }
    if (enemy01.Top == 650 || enemy02.Top == 650)
        nyawa -= 1;
    }
    if (nyawa <= 0)</pre>
        gameOver();
    }
```

4. Kode berikut berjalan ketika 'nyawa' sudah bernilai 0, kemudian melakukan *reset* keseluruhan variabel untuk memulai *game* yang baru.

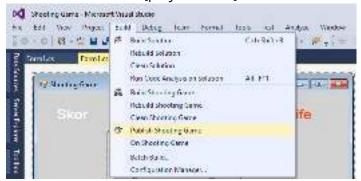
```
private void gameOver()
{
    timerPermainan.Enabled = false;
    MessageBox.Show("Skor Anda = " + skorPermainan + " Klik 'OK' untuk bermain
lagi");

skorPermainan = 0;
    nyawa = 3;
    labelSkor.Text = "Skor : 0";
    labelLife.Text = "Life : 3";
    enemy01.Top = -500;
    enemy02.Top = -1200;
    bullet.Top = -100;
    bullet.Left = -100;
    timerPermainan.Enabled = true;
}
```

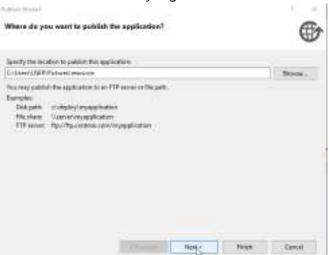
#### 1.2.6 Langkah 6-Mem-Publish Game yang Dirancang

Semua *game* perlu untuk dibagikan ke pemain lainnya untuk dimainkan oleh banyak orang, berikut cara mem-*publish game* dalam Visual Studio 2015.

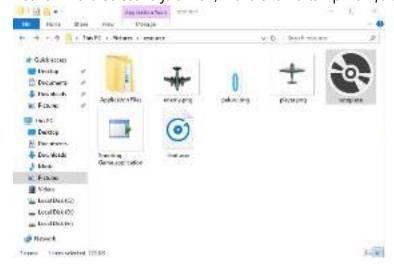
1. Klik 'Build -> Publish [project name]'.



2. Kemudian ikuti wizard yang disediakan.



3. Ketika wizard selesai dijalankan, maka akan ditampilkan folder seperti berikut.



#### 1.2.7 Langkah 7-Selamat Bermain dan Berbagi

Selamat bermain, *folder game* yang sudah dihasilkan dapat disalin dan dibagikan dengan teman secara langsung atau pun dengan melalui internet.

#### Referensi

- Alderman, N. (2015, Maret 9). Video game characters: the more real they get, the less we like them. Diambil kembali dari The gaming column with Naomi Alderman: https://www.theguardian.com/technology/2015/mar/09/game-character-redesigns-high-resolution
- Clingerman, G. W. (2008, Juli 5). *Using state to create game menus, display game screens and make your game characters come alive!* Diambil kembali dari The State of Things: http://www.xnadevelopment.com/tutorials/thestateofthings/thestateofthings.shtml
- Daniel. (2015, Agustus 21). *In Between A puzzle platformer game about life*. Diambil kembali dari Design: http://daniel.gentlymad.org/
- Eastwood, G. (2017, Januari 23). *How video game AI is changing the world*. Diambil kembali dari CIO: http://www.cio.com/article/3160106/artificial-intelligence/how-video-game-ai-is-changing-the-world.html
- Game Objects. (2017). Diambil kembali dari Game Design Novice: http://gamedesign.wikidot.com/game-object
- Gamefroot. (2017). What are game objects, and how do I place them? Diambil kembali dari Game Froot: http://gamefroot.com/knowledgebase/what-is-a-game-object/
- Koven, B. D. (2017). *FUN and flow.* Diambil kembali dari Deepfun: http://www.deepfun.com/funflow.htm
- LaMothe, A. (2017). *Designing Video Games*. Diambil kembali dari Dummies: http://www.dummies.com/programming/programming-games/designing-videogames/
- Mahinder, V. (2016, November 23). *2D vs 3D Animation -- Pros and Cons*. Diambil kembali dari Medium: https://medium.com/@vandhana.appiness/2d-vs-3d-animation-pros-and-cons-51daba16a4ec
- Microsoft. (2016, Maret 1). *Gears of War: Ultimate Edition*. Diambil kembali dari Gears of War: Ultimate Edition: https://gearsofwar.com/en-gb/games/gears-of-war-ultimate
- Microsoft. (2017). *Age of Empires*. Diambil kembali dari Age of Empires: https://www.ageofempires.com/
- Pluralsight. (2014, April 14). *How Animation for Games is Different from Animation for Movies*. Diambil kembali dari Plural Sight: https://www.pluralsight.com/blog/film-games/how-animation-for-games-is-different-from-animation-for-movies
- PotHix. (2012, November 30). *Gamedev Glossary: What Is the "Game Loop"?* Diambil kembali dari Game Development: https://gamedevelopment.tutsplus.com/articles/gamedev-glossary-what-is-the-game-loop--gamedev-2469

- PowerHouse Animation Studio, I. (2015, Juli 23). *The Role of Animation in Video Games*. Diambil kembali dari Slide Share: https://www.slideshare.net/powerhouseanima/role-of-animation-in-video-games
- Schreiber, I. (2009, Juli 20). *Level 7: Decision-Making and Flow Theory*. Diambil kembali dari Game Design Concepts: https://gamedesignconcepts.wordpress.com/2009/07/20/level-7-decision-making-and-flow-theory/

Seemann, G., & Bourg, D. M. (2004). Al for Game Developers. O'Reilly Media, Inc.

UNIVERSITAS

# Modul 2 - GameCapstone

# Game Functionality



### Didukung oleh:



## **2 GAME FUNCTIONALITY**

Game testing (pengujian permainan) merupakan bagian yang sangat penting dalam sebuah pengembangan game atau game development. Game testing adalah proses pengujian perangkat lunak untuk kontrol kualitas permainan video. Fungsi utama dari game testing adalah untuk menemukan dan mendokumentasikan bagian cacat dari game yang telah dikembangkan atau biasanya sering disebut dengan bugs. Pengujian game ini membutuhkan keahlian dalam bidang komputasi, analisis, dan keterampilan evaluasi (Bates, 2004).

Seiring dengan semakin rumitnya komponen permainan yang dikembangkan, para *publisher* biasanya akan menggunakan tenaga "Quality Assessment" atau "Quality Assurance" sebagai *game tester* untuk melakukan pengujian *game* ini.

#### **Tujuan Modul**

- 1. Memahami konsep pengujian dalam pengembangan game.
- 2. Mampu menambahkan desain art dan sound dalam permainan.
- 3. Mampu melakukan pemrograman tingkah laku karakter dan kecerdasan buatan dalam permainan.
- 4. Mampu mengimplementasikan tools dalam permainan.

#### 2.1 Ringkasan Pembelajaran

Sebelum membahas *game testing*, *game development* atau pengembangan *game* yang akan dilakukan adalah pendekatan mode *mid level* (level pertengahan) yang menggunakan *library* atau *framework* dengan tingkat yang lebih tinggi sehingga proses pembuatan *game* akan menjadi lebih sederhana. Pendekatan ini dimulai dengan mengenal aliran program sebuah *game*, bagaimana mengelola *resource* dari sebuah game seperti gambar, audio, video, menambahkan kontrol gerakan dan memberikan *gameplay* di dalamnya (Handoyo & Santoso, 2013).

Sebenarnya tidak ada metode standar untuk melakukan pengujian permainan karena biasanya metode pengujian akan dikembangkan oleh pengembang dan penerbit video game secara individual. Misalnya, metode pengujian untuk game MMORPG tentu akan berbeda dengan pengujian permainan dengan tema casual. Ada banyak metode pengujian yang dapat dilakukan seperti functionality testing, compliance testing, compatibility testing, compatibility testing, localization testing, soak testing, beta testing, regression testing, load testing, dan mobile game testing.

Function testing adalah metode pengujian yang paling sering digunakan karena metode ini tidak memerlukan banyak sisi teknis. Pengujian ini hanya melibatkan user interface sebagai antarmuka pengguna, masalah stabilitas (frame dan screen), masalah game mechanic, dan aset game seperti gambar, animasi, suara, dan lainnya.

Jadi, game function pada modul ini membahas bagaimana proses pengembangan game mulai dari gameplay, aset gambar hingga pengujian game dengan function testing menggunakan framework sederhana. Ada beberapa framework yang dapat diperoleh secara gratis seperti XNA, SharpDX, DirectX Tool Kit, SlimDX, Cocos 2D, MonoGame, dan lain sebagainya. Framework yang digunakan dalam modul ini adalah MonoGame.

#### 2.1.1 Apa itu MonoGame?

MonoGame adalah sebuah *framework open source* yang dibuat menggunakan Microsoft XNA 4.0. Tujuan dari pembuatan *framework* ini adalah untuk memberikan jalan bagi para *developer* XNA agar dapat memindahkan *game* mereka ke dalam iOS, Android, Mac OS, Linux, dan Windows 8. Sampai saat ini MonoGame Framework telah mencapai versi yang ketiga dan masih berada dalam tahap pengembangan lebih lanjut untuk membuatnya lebih sempurna.

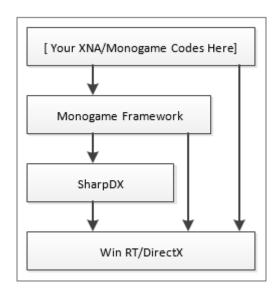


Gambar 2. 1 Logo MonoGame

Terdapat beberapa teknologi yang digunakan MonoGame sehingga membuatnya *cross-platform*, yaitu (Walker, 2012):

- 1. OpenTK: OpenTK adalah sebuah *library* grafik yang menyediakan bantuan untuk penggunaan OpenGL, OpenCL, dan OpenAL.
- 2. SharpDX: SharpDX adalah sebuah *library* .NET yang bertujuan membungkus API DirectX untuk keperluan pembuatan *game*. Dengan demikian, para *programmer* yang tidak memiliki dasar C++, tidak perlu bersusah payah untuk mempelajari C++ terlebih dahulu untuk membuat *game*.
- 3. Lidgren Network: Lidgren Network adalah sebuah *library* jaringan di atas .NET *framework* yang menyediakan bantuan koneksi dari *client* ke *server*.

MonoGame didesain berdasarkan dari XNA Framework yang telah dibuat oleh Microsoft. XNA Framework adalah sebuah *framework* yang dibuat oleh Microsoft untuk pengembangan *game* pada Windows Phone 7, XBOX 360, dan desktop. MonoGame dibuat semirip mungkin dengan XNA Framework karena *framework* ini ditujukan pula sebagai *framework* alternatif bagi *developer* XNA yang saat ini sedang mengalami berbagai kesulitan dikarenakan putusnya dukungan Microsoft pada XNA Framework ini.



Gambar 2. 2 Arsitektur MonoGame (Handoyo & Santoso, 2013)

Dapat dilihat bahwa SharpDX membungkus proses teknis yang terdapat di dalam DirectX. MonoGame Framework menggunakan SharpDX ini untuk membuat *framework* yang serupa dengan XNA Framework. Dengan demikian, MonoGame Framework memiliki dua buah keuntungan sekaligus. Yang pertama, MonoGame Framework dapat digunakan oleh para *developer* pemula untuk mengembangkan proyek *game* miliknya. Yang kedua, MonoGame Framework akan dapat digunakan dengan sangat mudah oleh para *developer* yang dulunya telah memakai XNA Framework. Beberapa *namespace* utama yang dimiliki oleh MonoGame (dan XNA) adalah sebagai berikut (MSDN, 2017):

- Namespace Framework: berisi fungsionalitas dasar *game* seperti *timer* dan *game loop*.
- Namespace Framework.Audio: berisi API untuk memanipulasi audio.
- Namespace Framework.Content: berisi komponen untuk pengolahan resource game.
- Namespace Framework. Design: berisi *library* yang berfungsi untuk mengubah nilai dari satu tipe ke tipe yang lain.
- Namespace Framework.GamerService: berisi API untuk mengatur komunikasi antar *user*, data *user*, dan sebagainya.
- Namespace Framework.Graphics: berisi API yang dapat menggunakan *hardware* untuk mempercepat penampilan objek 3D.
- Namespace Framework.Input: berisi API untuk berbagai macam tipe *input* dari *user* (*keyboard, mouse, controller*).
- Namespace Framework.Media: berisi API untuk menjalankan berkas musik dan gambar.
- Namespace Framework.Net: berisi dukungan untuk XBOX LIVE, *multiplayer*, dan *networking* lainnya.
- Namespace Framework.Storage: berisi *class* yang digunakan untuk menulis dan membaca ke dan dari media penyimpanan berkas.

Terdapat beberapa kelebihan utama yang terdapat pada MonoGame:

- 1. MonoGame dapat digunakan secara gratis tanpa biaya dan open source.
- 2. Produk game yang dibuat dapat dimainkan di berbagai platform yang berbeda (iOS, Android, Windows 8, Mac OS, Linux).
- 3. Salah satu IDE yang dapat digunakan untuk mengembangkan *game* dengan MonoGame adalah Visual Studio yang juga dapat diunduh secara gratis.

#### 2.1.2 Instalasi MonoGame

Sebelum dapat membuat *game* di Windows menggunakan MonoGame, pastikan bahwa Integrated Development Environment (IDE) sudah dipersiapkan terlebih dahulu. Ikuti dan lakukan langkah-langkah berikut untuk mempersiapkan XNA Development dengan MonoGame:

- Pasang Visual Studio Community 2017
   Mengapa membutuhkan Visual Studio Community 2017? Terdapat fitur XNA yang disebut Content Pipeline, sebuah pre-compiler yang berfungsi untuk mempersiapkan aset grafis dan audio saat program dijalankan (runtime) di XNA. Fitur ini tidak ada pada MonoGame. Pengguna dapat menggunakan Visual Studio Community 2017 untuk memanfaatkan template XNA Framework untuk mengompilasi grafis ke dalam berkas .xnb untuk disertakan dalam proyek Windows (panduan ini menggunakan Visual Studio Community 2017). Visual Studio dapat diunduh secara gratis di https://www.visualstudio.com/downloads/.
- 2. Pasang MonoGame SDK melalui <a href="http://www.monogame.net/downloads/">http://www.monogame.net/downloads/</a> dan pilih, MonoGame for Visual Studio.

#### 2.1.3 Tutorial MonoGame

MonoGame adalah *tools* gratis yang dapat digunakan oleh pengembang untuk membuat game di Windows dan Windows Phone dapat berjalan di sistem lain seperti OS X, Linux, iOS, Android, PlayStation Mobile, Nintendo Switch dan konsol OUYA berkat implementasi antarmuka pemrograman Microsoft XNA 4. Berikut beberapa tautan tutorial yang dapat dipelajari seperti menambahkan gambar, audio, video melalui Pipeline (*front-end* GUI Editor) (Whitaker, n.d.) atau menuliskan kode pemrograman dasar untuk menampilkan objek 2 dimensi dan memberikan pengaturan di dalamnya (Schulze, 2014).

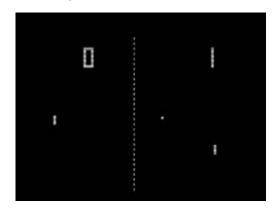
- 1. http://www.monogame.net/documentation/?page=Tutorials
- 2. <a href="http://rbwhitaker.wikidot.com/monogame-tutorials">http://rbwhitaker.wikidot.com/monogame-tutorials</a>
- 3. <a href="https://en.wikibooks.org/wiki/Game\_Creation\_with\_XNA/2D\_Development/Sprites">https://en.wikibooks.org/wiki/Game\_Creation\_with\_XNA/2D\_Development/Sprites</a>

#### 2.2 Video Tutorial URL

Penerapan game function pada modul ini membahas bagaimana proses pengembangan game mulai dari gameplay, aset gambar hingga pengujian game dengan function testing menggunakan framework sederhana, MonoGame. Contoh game yang akan dibuat adalah Pong Game.

Pong adalah permainan video generasi pertama yang dirilis sebagai permainan arkade yang dioperasikan dengan koin yang dikembangkan oleh Atari Inc. pada tanggal 29 November, 1972. Pong didasari dari permainan atau olahraga tenis meja (pingpong), nama permainan ini berasal dari suara yang dihasilkan ketika memukul bola pingpong. Kata Pong telah didaftarkan sebagai merek dari Atari Interactive (Winter, 2007).

Pong adalah permainan video olahraga dua dimensi yang menyimulasikan sebuah permainan tenis meja. Sang pemain menggerakkan sebuah *bat* yang bergerak naik-turun, permainan ini dapat dimainkan dengan komputer atau dua orang yang menggerakkan *bat* di sebelah kiri. Pemain memukul bola bolak-balik. Untuk memenangkan permainan, sang pemain harus memukul bola tersebut sampai sang lawan tidak bisa memukul bola tersebut (Sellers, 2001).



Gambar 2. 3 Pong Game oleh Atari (Winter, 2007)

Pong yang dibuat akan diberikan nama Ping Pong Battle dengan ketentuan sebagai berikut:

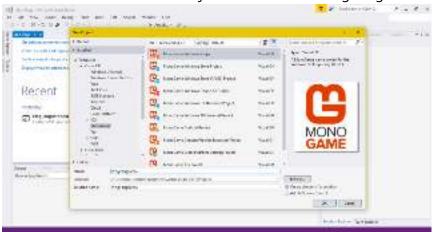
- 1. Aset gambar yang akan digunakan sebanyak tiga buah, yaitu *background* sebagai latar belakang atau meja permainan, *bat* sebagai papan ping pong, dan *ball* sebagai bola.
- 2. Player 1 di bagian kiri dimainkan dengan menggunakan tombol W dan S. Sedangkan, Player 2 di bagian kanan dimainkan dengan menggunakan tombol Atas dan Bawah.
- 3. Ketika permainan dimulai, bola akan terletak di tengah meja dan dapat dimulai dengan menekan tombol spasi.
- 4. Kecepatan bola akan terus bertambah setiap detiknya dan jika salah satu pemain gagal menangkap bola tersebut maka lawan akan mendapatkan poin skor yang tertera di atas layar.

Tautan Video Tutorial: <a href="https://youtu.be/lUEcdw-BOBM">https://youtu.be/lUEcdw-BOBM</a>.

#### 2.2.1 Langkah 1-Membuat Project MonoGame Baru

Sebelumnya, pastikan dulu bahwa Visual Studio sudah terinstal di komputer atau laptop (panduan ini menggunakan Visual Studio Community 2017). Visual Studio dapat diunduh secara gratis di https://www.visualstudio.com/downloads/.

- 1. Instal MonoGame SDK melalui <a href="http://www.monogame.net/downloads/">http://www.monogame.net/downloads/</a> dan pilih, MonoGame for Visual Studio.
- 2. Buka Visual Studio Community 2017. Lalu, klik Create new project untuk membuat *project* baru.
- 3. Pilih MonoGame Windows Project. Berikan nama PingPongBattle.



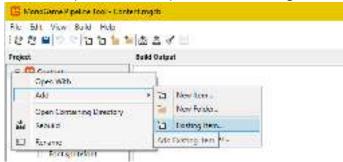
#### 2.2.2 Langkah 2-Menggunakan MonoGame Pipeline

MonoGame Pipeline adalah fitur MonoGame yang memungkinkan pengguna untuk dapat menggugah aset seperti gambar, audio, video, font, dan lainnya untuk dapat ditampilkan ke dalam Visual Studio.

1. Buka folder Contents dan klik 2x file Content.mgcb untuk mengakses MonoGame Pipeline.



2. Klik kanan pada Content, pilih Add > Existing Item untuk mengunggah aset gambar.



3. Pilih Background, Bat, dan Ball yang telah disiapkan.



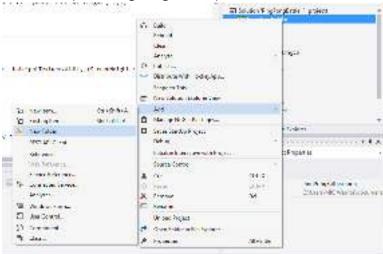
4. Pilih Game1.cs untuk menambahkan kode pertama (sisanya dilanjutkan pada langkah terakhir). Tuliskan kode di bawah ini tepat di bawah kode SpriteBatch spriteBatch;

```
public static int ScreenWidth;
public static int ScreenHeight;
public static Random Random;
```

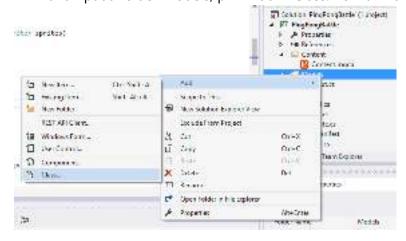
#### 2.2.3 Langkah 3-Folder Models dan Class Input.cs

Folder Models ini berfungsi untuk memberikan nilai masukkan kontrol gerakan *bat* sebagai papan pingpong.

1. Klik kanan pada PingPongBattle, pilih Add > New Folder. Berikan nama Models.



2. Klik kanan pada folder Models, pilih Add > Class. Berikan nama Input.cs.



3. Tambahkan kode skrip berikut yang berfungsi untuk memberikan gerakan naik turun pada *bat* ketika nanti dimainkan.

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Input;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace PingPongBattle.Models
{
    public class Input
    {
        public Keys Up;
        public Keys Down;
    }
}
```

#### 2.2.4 Langkah 4- Folder Sprites dan Class Sprite.cs

Folder Sprites ini berfungsi untuk menambahkan class Sprite.cs sebagai kode untuk pengaturan pada layar permainan, class Bat.cs sebagai kode untuk pengaturan papan ping pong, dan class Ball.cs sebagai kode untuk pengatura bola.

- 1. Klik kanan pada PingPongBattle, pilih Add > New Folder. Berikan nama Sprites.
- 2. Klik kanan pada folder Sprites, pilih Add > Class. Berikan nama Sprite.cs.
- 3. Tambahkan kode berikut.

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using PingPongBattle.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace PingPongBattle.Sprites
   public class Sprite
        protected Texture2D texture;
        public Vector2 Position;
        public Vector2 Velocity;
        public float Speed;
        public Input Input;
        public Rectangle Rectangle
            get
                return new Rectangle((int)Position.X, (int)Position.Y,
_texture.Width, _texture.Height);
```

```
}
        public Sprite(Texture2D texture)
            _texture = texture;
        public virtual void Update(GameTime gameTime, List<Sprite> sprites)
        }
        public virtual void Draw(SpriteBatch spriteBatch)
            spriteBatch.Draw(_texture, Position, Color.White);
        #region Collision
        protected bool IsTouchingLeft(Sprite sprite)
            return this.Rectangle.Right + this.Velocity.X >
sprite.Rectangle.Left &&
                   this.Rectangle.Left < sprite.Rectangle.Left &&</pre>
                   this.Rectangle.Bottom > sprite.Rectangle.Top &&
                   this.Rectangle.Top < sprite.Rectangle.Bottom;</pre>
        }
        protected bool IsTouchingRight(Sprite sprite)
            return this.Rectangle.Left + this.Velocity.X <</pre>
sprite.Rectangle.Right &&
                   this.Rectangle.Right > sprite.Rectangle.Right &&
                   this.Rectangle.Bottom > sprite.Rectangle.Top &&
                   this.Rectangle.Top < sprite.Rectangle.Bottom;</pre>
        }
        protected bool IsTouchingTop(Sprite sprite)
            return this.Rectangle.Bottom + this.Velocity.Y >
sprite.Rectangle.Top &&
                   this.Rectangle.Top < sprite.Rectangle.Top &&</pre>
                   this.Rectangle.Right > sprite.Rectangle.Left &&
                   this.Rectangle.Left < sprite.Rectangle.Right;</pre>
        }
        protected bool IsTouchingBottom(Sprite sprite)
            return this.Rectangle.Top + this.Velocity.Y <</pre>
sprite.Rectangle.Bottom &&
                   this.Rectangle.Bottom > sprite.Rectangle.Bottom &&
                   this.Rectangle.Right < sprite.Rectangle.Left &&</pre>
                   this.Rectangle.Left > sprite.Rectangle.Right;
        #endregion
    }
}
```

#### 2.2.5 Langkah 5- Menambahkan Class Score.cs

Class Score.cs berfungsi untuk pengaturan nilai skor antara Player 1 dan Player 2 seperti menginisiasi tampilan font untuk skor pada layar dan posisi koordinatnya.

- 1. Klik kanan pada PingPongBattle, lalu pilih Add > Class. Berikan nama Score.cs.
- 2. Tambahkan kode berikut.

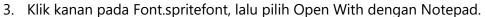
```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace PingPongBattle
   public class Score
        public int Score1;
        public int Score2;
        private SpriteFont font;
        public Score(SpriteFont font)
            font = font;
        public void Draw(SpriteBatch spriteBatch)
            spriteBatch.DrawString(_font, Score1.ToString(), new Vector2(320,
70), Color.White);
            spriteBatch.DrawString(_font, Score2.ToString(), new Vector2(430,
70), Color.White);
}
```

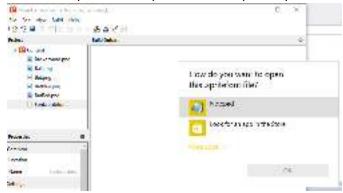
#### 2.2.6 Langkah 6- Menambahkan Font

Font digunakan untuk menampilkan sprite Font pada layar permainan berupa skor.

- 1. Buka MonoGame Pipeline. Klik kanan pada Content, pilih Add > New Item.
- 2. Pilih SpriteFont Description dan berikan nama Font. Lalu, klik OK.







4. Ubahlah ukuran huruf menjadi 48.

```
Fig. 12th forms the last characters which seems as all description of a form, and will as read by the sear reason content of the form the first content of the content of the search of the form in your gase, and to change the characters which are evaluable to dreadth.—**Content of the form in your gase, and to change the characters which are evaluable to dreadth.—**Content of the form in your gase, and to change the characters which are evaluable to dreadth.—**Content of the form in your gase, and to change the characters which are evaluable to characters of the form of the search of the characters of the form of the search of the characters of the form of the form of the characters of the form of the characters.

**A formism and of the state of the form of the characters of the characters of the characters of the characters.

**A character of the characters of the form of the characters of the characters of the characters.

**Content of the characters of the characters
```

# 2.2.7 Langkah 7- Menambahkan Class Bat.cs

Class Bat.cs berfungsi mengatur objek gambar Bat sebagai papan ping pong seperti menampilkan Bat di layar permainan, kecepatan Bat, kontrol gerakan, dan posisi koordinatnya.

```
using Microsoft.Xna.Framework.Graphics;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Input;
namespace PingPongBattle.Sprites
{
    public class Bat : Sprite
        public Bat(Texture2D texture)
            : base(texture)
            Speed = 5f;
        public override void Update(GameTime gameTime, List<Sprite> sprites)
            if (Input == null)
```

# 2.2.8 Langkah 8- Menambahkan Class Ball.cs

Class Ball.cs berfungsi mengatur objek gambar Ball sebagai bola seperti menampilkan Ball di layar permainan, kecepatan bola, kontrol gerakan, arah gerakan bola jika menabrak Bat, dan posisi koordinatnya.

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace PingPongBattle.Sprites
   public class Ball : Sprite
        private float _timer = Of; //Increment the speed over time
        private Vector2? _startPosition = null;
        private float? _startSpeed;
        private bool _isPlaying;
        public Score Score;
        public int SpeedIncrementSpan = 10; //How often the speed with increment
        public Ball(Texture2D texture)
            :base(texture)
            Speed = 3f;
        }
        public override void Update(GameTime gameTime, List<Sprite> sprites)
            if (_startPosition == null)
                _startPosition = Position;
                startSpeed = Speed;
```

```
Restart();
            if (Keyboard.GetState().IsKeyDown(Keys.Space))
                _isPlaying = true;
            if (!_isPlaying)
                return;
            _timer += (float)gameTime.ElapsedGameTime.TotalSeconds;
            if (_timer > SpeedIncrementSpan)
                Speed++;
                _{timer} = 0;
            foreach (var sprite in sprites)
                if (sprite == this)
                    continue;
                if (this.Velocity.X > 0 && this.IsTouchingLeft(sprite))
                     this.Velocity.X = -this.Velocity.X;
                if (this.Velocity.X < 0 && this.IsTouchingRight(sprite))</pre>
                    this.Velocity.X = -this.Velocity.X;
                if (this.Velocity.Y > 0 && this.IsTouchingTop(sprite))
                    this.Velocity.Y = -this.Velocity.Y;
                if (this.Velocity.Y < 0 && this.IsTouchingBottom(sprite))</pre>
                    this.Velocity.Y = -this.Velocity.Y;
            if (Position.Y <= 0 || Position.Y + texture.Height >=
Game1.ScreenHeight)
                Velocity.Y = -Velocity.Y;
            if (Position.X <= 0)</pre>
            {
                Score.Score2++;
                Restart();
            }
            if (Position.X + _texture.Width >= Game1.ScreenWidth)
                Score.Score1++;
                Restart();
            Position += Velocity * Speed;
        }
        public void Restart()
            var direction = Game1.Random.Next(0, 4);
            switch (direction)
                case 0:
                    Velocity = new Vector2(1, 1);
                    break;
                case 1:
                    Velocity = new Vector2(1, -1);
                    break;
```

```
case 2:
    Velocity = new Vector2(-1, -1);
    break;
case 3:
    Velocity = new Vector2(-1, 1);
    break;
}

Position = (Vector2)_startPosition;
Speed = (float)_startSpeed;
    _timer = 0;
    _isPlaying = false;
}
}
```

# 2.2.9 Langkah 9- Menambahkan Kode di Game1.cs

Class Game1.cs berfungsi mengatur *gameplay* secara keseluruhan berdasarkan class yang telah dibuat sebelumnya. Gameplay yang ada seperti menginisiasi layar permainan berdasarkan Sprite.cs, memanggil objek gambar dan font, menetapkan posisi koordinat dan kontrol gerakan untuk Bat dan Ball, dan pengaturan nilai skor.

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using PingPongBattle.Models;
using PingPongBattle.Sprites;
using System;
using System.Collections.Generic;
namespace PingPongBattle
    /// <summary>
   /// This is the main type for your game.
   /// </summary>
   public class Game1 : Game
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;
        public static int ScreenWidth;
        public static int ScreenHeight;
        public static Random Random;
        private Score _score;
        private List<Sprite> _sprites;
        public Game1()
            graphics = new GraphicsDeviceManager(this);
            Content.RootDirectory = "Content";
        }
        /// <summary>
```

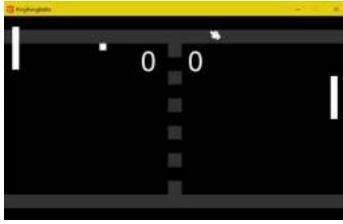
```
/// Allows the game to perform any initialization it needs to before
starting to run.
        /// This is where it can query for any required services and load any non-
graphic
        /// related content. Calling base. Initialize will enumerate through any
components
        /// and initialize them as well.
        /// </summary>
        protected override void Initialize()
            // TODO: Add your initialization logic here
            ScreenWidth = graphics.PreferredBackBufferWidth;
            ScreenHeight = graphics.PreferredBackBufferHeight;
            Random = new Random();
            base.Initialize();
        }
        /// <summary>
        /// LoadContent will be called once per game and is the place to load
        /// all of your content.
        /// </summary>
        protected override void LoadContent()
        {
            // Create a new SpriteBatch, which can be used to draw textures.
            spriteBatch = new SpriteBatch(GraphicsDevice);
            var batleftTexture = Content.Load<Texture2D>("BatRed");
            var batrightTexture = Content.Load<Texture2D>("BatBlue");
            var ballTexture = Content.Load<Texture2D>("Ball");
            score = new Score(Content.Load<SpriteFont>("Font"));
            _sprites = new List<Sprite>()
                new Sprite(Content.Load<Texture2D>("Background")),
                new Bat(batleftTexture)
                    Position = new Vector2(20, (ScreenHeight / 2) -
(batleftTexture.Height / 2)),
                    Input = new Input()
                        Up = Keys.W,
                        Down = Keys.S,
                },
                new Bat(batrightTexture)
                    Position = new Vector2(ScreenWidth - 20 - batrightTexture.Width,
(ScreenHeight / 2) - (batrightTexture.Height)),
                    Input = new Input()
                        Up = Keys.Up,
                        Down = Keys.Down,
                },
                new Ball(ballTexture)
                    Position = new Vector2((ScreenWidth / 2) - (ballTexture.Width /
2), (ScreenHeight / 2) - (ballTexture.Height / 2)),
                    Score = _score,
```

```
};
            // TODO: use this.Content to load your game content here
        }
        /// <summary>
        /// UnloadContent will be called once per game and is the place to unload
        /// game-specific content.
        /// </summary>
        protected override void UnloadContent()
            // TODO: Unload any non ContentManager content here
        }
        /// <summary>
        /// Allows the game to run logic such as updating the world,
        /// checking for collisions, gathering input, and playing audio.
        /// </summary>
        /// <param name="gameTime">Provides a snapshot of timing values.</param>
        protected override void Update(GameTime gameTime)
            foreach(var sprite in _sprites)
                sprite.Update(gameTime, _sprites);
            // TODO: Add your update logic here
            base.Update(gameTime);
        }
        /// <summary>
        /// This is called when the game should draw itself.
        /// </summary>
        /// <param name="gameTime">Provides a snapshot of timing values.</param>
        protected override void Draw(GameTime gameTime)
            GraphicsDevice.Clear(Color.CornflowerBlue);
            spriteBatch.Begin();
            foreach (var sprite in _sprites)
                sprite.Draw(spriteBatch);
                score.Draw(spriteBatch);
            spriteBatch.End();
            // TODO: Add your drawing code here
            base.Draw(gameTime);
        }
    }
}
```

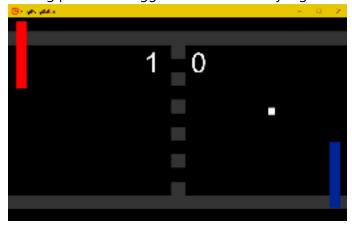
# 2.2.10 Hasil Pengujian Game

Pengujian *game* akan dilakukan dengan metode *function testing* untuk melihat *bugs* dalam *game* yang dikembangkan mulai dari apakah *gameplay* berjalan sesuai dengan skenario, apakah ada gambar yang tidak termuat, apakah ada gerakan bola berhasil memantul ke segala arah ketika bertabrakan dengan layar atau Bat, dan lainnya.

- 1. Simpan terlebih dahulu semua pembaruan kode pemrograman yang telah dilakukan pada masing-masing class dengan Ctrl+S.
- 2. Klik tombol Start di bagian atas menu untuk menjalankan game.



3. Berikan modifikasi pada kode pemrograman yang ada seperti membuat masingmasing pemain menggunakan warna Bat yang berbeda.



#### Referensi

- Bates, B. (2004). Game Design 2nd Edition. Boston: Thomson Course Technology.
- Handoyo, E. D., & Santoso, S. (2013). Monogame Framework sebagai Salah Satu Framework Alternatif pada Mata Kuliah Pemrograman Game. *Setisi 2013* (hal. 43-46). Bandung: Seminar Teknik Informatika dan Sistem Informasi.
- MSDN. (2017, May 23). XNA Game Studio 4.0 Refresh. Diambil kembali dari Microsoft Developer Network: https://msdn.microsoft.com/en-us/library/bb200104.aspx
- Schulze, S. (2014, March 20). *Game Creation with XNA/2D Development/Sprites*. Diambil kembali dari WikiBooks: https://en.wikibooks.org/wiki/Game\_Creation\_with\_XNA/2D\_Development/Sprites
- Sellers, J. (2001). *Arcade Fever The Fan's Guide To The Golden Age Of Video Games*. Philadelphia: Running Press.
- Sharma, S. (2010). *Nova Air Hockey Tables HT 3014ES*. Diambil kembali dari Indiamart: https://www.indiamart.com/proddetail/nova-air-hockey-tables-ht-3014es-10545064130.html
- Team, M. (2009). *What is MonoGame*. Diambil kembali dari MonoGame: http://www.monogame.net/documentation/
- Walker, T. E. (2012, December 04). Windows 8 Game Development using C#, XNA and MonoGame 3.0: Building a Shooter Game Walkthrough Part 1: Overview, Installation, MonoGame 3.0 Project Creation. Diambil kembali dari Microsoft Developer: https://blogs.msdn.microsoft.com/tarawalker/2012/12/04/windows-8-game-development-using-c-xna-and-monogame-3-0-building-a-shooter-game-walkthrough-part-1-overview-installation-monogame-3-0-project-creation/
- Whitaker, R. (t.thn.). *Monogame 2D Tutorials*. Diambil kembali dari RB Whitaker's Wiki: http://rbwhitaker.wikidot.com/monogame-2d-tutorials
- Winter, D. (2007, May 26). *Atari PONG The first steps -*. Diambil kembali dari pong-story.com: http://www.pong-story.com/atpong1.htm

UNIVERSITAS

# Modul 3 - GameCapstone

# Pengembangan Game Story Berbasis Sandbox



# Didukung oleh:



# 3 PENGEMBANGAN GAME BERBASIS SANDBOX

Sandbox berbeda dengan open world. Open world adalah istilah untuk permainan video game yang memungkinkan pemain menggerakkan karakternya dengan alur permainannya yang cenderung menyebar, karena pemain tidak harus langsung menyelesaikan objektif permainan, salah satu contohnya seperti permainan GTA (Grand Theft Auto). Sedangkan, sandbox memiliki alur permainan yang linier. Karakter pemain diberikan kemampuan untuk dapat melakukan apa saja dengan sumber daya yang telah disediakan. Sandbox juga memiliki konsep pembuatan dan perancangan suatu objek pada gameplay permainan dan dimainkan dalam lingkungan dunia terbuka seperti Minecraft (Booker, 2008).

# **Tujuan Modul**

- 1. Mampu memahami enam tahapan dalam pembuatan *game story*.
- 2. Mampu membuat game story untuk permainan yang dirancang.

# 3.1 Ringkasan Pembelajaran

Jalan cerita dalam sebuah permainan merupakan pengalaman yang harus dinikmati oleh setiap pemain. *Game story* menjadi salah satu komposisi dalam pengembangan permainan yang penting untuk dilakukan, karena setiap jalan cerita akan menuju pada *gameplay* yang menjelaskan kepada pemain apa saja yang harus dilakukan untuk menyelesaikan permainan tersebut (Schell, 2008).

Lalu, bagaimana caranya untuk dapat membuat *game story* yang baik? Pertama, sebagai pengembang *game* yang akan menuliskan *game story*, maka orang tersebut haruslah seorang *gamer* (pemain game) karena hasilnya akan jadi jauh lebih baik dan mendalami. Pengembang tersebut akan mengerti bagaimana *game story* ini mengalir ke arah *gameplay* yang menjelaskan interaksi *game* kepada pemain. Pengembang juga akan memahami siapa saja pahlawan dan penjahat yang terlibat, objektif yang dikerjakan dan dukungan lainnya (Danny, 2015). Kedua, menentukan plot. Plot penting karena menjadi sumber konflik atau pemicu dalam *game story* yang dirancang menjadi sesuatu yang harus dihadapi oleh pemain. Plot akan membuat permainan menjadi lebih interaktif.

# 3.1.1 Pengenalan Minecraft

Minecraft adalah permainan bak pasir yang diciptakan oleh pengembang asal Swedia, Markus "Notch" Persson, dan sekarang dikembangkan oleh perusahaannya, Mojang. Pada tahun 2011, Jens Bergensten, juga dikenal dengan nama samaran Jeb, mengambil alih kendali Minecraft. Minecraft difokuskan pada kreativitas dan pembangunan, yang memungkinkan pemain untuk membangun konstruksi dari kubus bertekstur dalam dunia 3D (AB, 2017).

Minecraft mempunyai dua mode permainan yakni Survival dan Creative. Mode Survival mengharuskan pemain untuk mendapatkan sumber daya sendiri dan memiliki poin nyawa dan lapar, pastinya pemain harus menjadi survivor, bertahan hidup, mencari sumber daya alam dan membuat bangunan dengan hasil sumber daya alam yang kita dapatkan. Lalu ada mode Creative, dalam mode ini pemain memiliki sumber daya yang tak terbatas, kemampuan untuk terbang, dan tidak ada poin nyawa maupun lapar. Jadi di mode ini pemain hanya mengandalkan kreativitas untuk membangun sesuatu yang imajinatif tanpa harus mencari sumber daya alam lagi.

Perhatikan, dalam praktik di modul dan video, Minecraft yang digunakan adalah Minecraft: Education Edition atau versi edukasi dari Minecraft. Untuk memainkannya pengguna harus memiliki akun Office 365 dari Microsoft. Unduh di sini <a href="https://education.minecraft.net/get-started/">https://education.minecraft.net/get-started/</a>.



Gambar 2. 1 Minecraft: Education Edition

# 3.1.2 Tahapan Pembuatan Game Story

Setiap permainan biasanya memiliki jalan cerita atau biasa disebut dengan *game story*. Meskipun beberapa *game* ada yang sepertinya tidak memiliki cerita, tetapi jika pemain melihat lebih jeli, maka akan menemui bahwa sebenarnya permainan tersebut memiliki jalan cerita yang tersembunyi. *Game story* sangat penting dalam sebuah permainan untuk mendukung penyampaian *gameplay* yang baik. Dalam *game* biasanya ada pahlawan, penjahat dan konflik. *Game story* memberitahukan pemain apa yang perlu mereka pahami sebelum memulai permainan dan bagaimana apa saja objektif yang harus diselesaikan (Derk, 2012).

Berikut langkah-langkah untuk pembuatan game story dalam sebuah pengembangan game:

## 1. Langkah 1: Menciptakan Dunia

Permainan akan lebih baik jika dimulai dengan cerita mulai dari pengenalan dunia permainan yang akan dimainkan. Jelaskan komponen geografi dunia *game* tersebut.

#### 2. Langkah 2: Menciptakan Karakter

Karakter adalah pemain utama dalam sebuah permainan. Jelaskan kepribadian karakter dan bagaimana bentuknya. Tambahkan juga pengenalan karakter untuk musuh-musuh yang nantinya akan dihadapi oleh karakter pahlawan.

## 3. Langkah 3: Menuliskan Storyline dan Gameplay

Storyline adalah jalan cerita dari permainan. Storyline akan dikaitkan dengan cerita dunia dan karakter yang sebelumnya telah dibangun. Ceritakan apa yang menjadi dasar karakter memulai petualangannya. Gameplay adalah aturan main yang diterapkan dalam sebuah permainan. Mulai dari objektif apa saja yang harus diselesaikan oleh karakter dan bagaimana ia dapat memulai dan mengakhiri permainan.

#### 4. Langkah 4: Menentukan Plot

Plot adalah sumber konflik. Ceritakan faktor apa yang menjadi dasar dari terjadinya konflik tersebut sehingga pemain harus melibatkan emosi dengan karakter dalam menyelesaikan permainan.

## 5. Langkah 5: Merancang Dunia dan Elemen Penyusun Game

Rancanglah bangun dunia sesuai dengan skenario storyline dan gameplay yang telah ditentukan bersama dengan elemen penyusun game seperti audio, cutscene, storyboard dan lainnya.

# 6. Langkah 6: Implementasi Game

Cobalah untuk memainkan dunia permainan yang telah dibuat untuk merasakan kekurangan yang perlu diperbaiki atau meningkatkan kelebihan yang sudah ada mulai dari karakter, *gameplay*, *storyline*.

# 3.2 Video Tutorial URL

Game story dalam Minecraft: Education Edition ini menggunakan bioma Jungle yang dapat diunduh di <a href="https://education.minecraft.net/class-resources/worlds/">https://education.minecraft.net/class-resources/worlds/</a>. Pertama, sebagai pemain yang menggerakkan Alex (karakter Minecraft) akan memainkan dunia Minecraft dalam mode kreatif untuk merancang bangunan dan menyusun blok sesuai dengan skenario yang telah dibuat. Setelah selesai, barulah dunia Minecraft dimainkan dalam mode bertahan hidup dan pemain harus menyelesaikan misi yang diberikan secepat mungkin.

Tautan video tutorial: <a href="https://youtu.be/ueeggOT7f3o">https://youtu.be/ueeggOT7f3o</a>.

# Storyline: Alex, Ranger Muda Hutang Lindung Minecraftopia

Alex adalah seorang penjaga hutan yang baru saja dilantik oleh kesatuan Pusat Ranger (Ranger adalah nama profesi dari penjaga hutan). Ia ditugaskan untuk mengawasi Hutan Lindung di Minecraftopia (dunia Minecraft khusus dengan bioma hutan) di sektor X. Alex harus melaporkan keadaan hutan setiap harinya, mengawasi area hutan agar tidak terjadi kebakaran hutan saat sedang musim kemarau seperti saat ini dan mencegah adanya penebangan hutan liar oleh para kontraktor. Lalu, semua area hutan tersebut harus dipetakan.

Sayangnya, di hari pertama bekerja pada akhir pekan, Alex menemukan beberapa penjahat yakni para kontraktor usil yang berencana akan menebang pohon-pohon dan membakar hutan serta menginginkan kehancuran alam Minecraftopia. Alex hanya sendirian dan harus melawan para kontraktor yang terdiri dari *hostile mob* ras Vindicator dan Evoker bahkan mereka membawa Ghast dari dunia Ender (dunai alam baka Minecraft) untuk membakar hutan. Dimulailah tugas Alex untuk menyelamatkan hutan demi menjaga keseimbangan alam Minecarftopia.



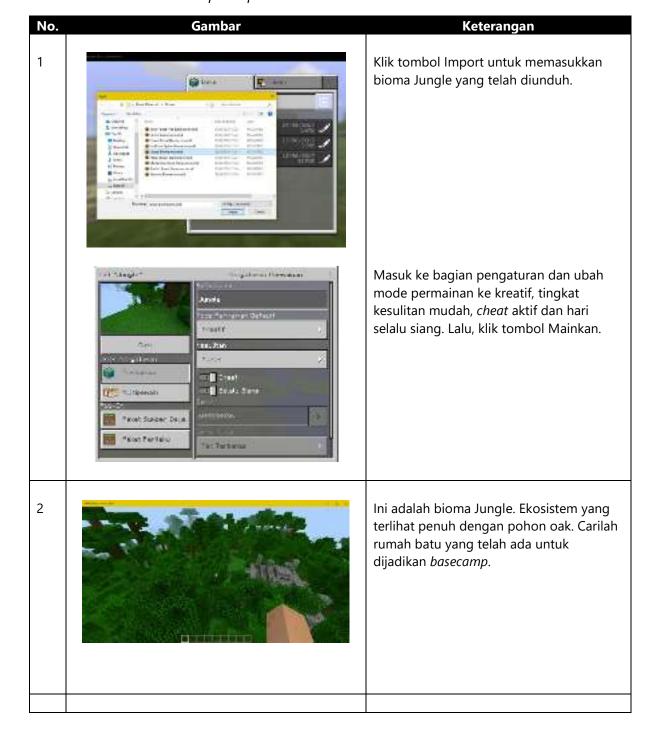
Gambar 2. 2 Alex Si Ranger Hutang Lindung Minecraftopia

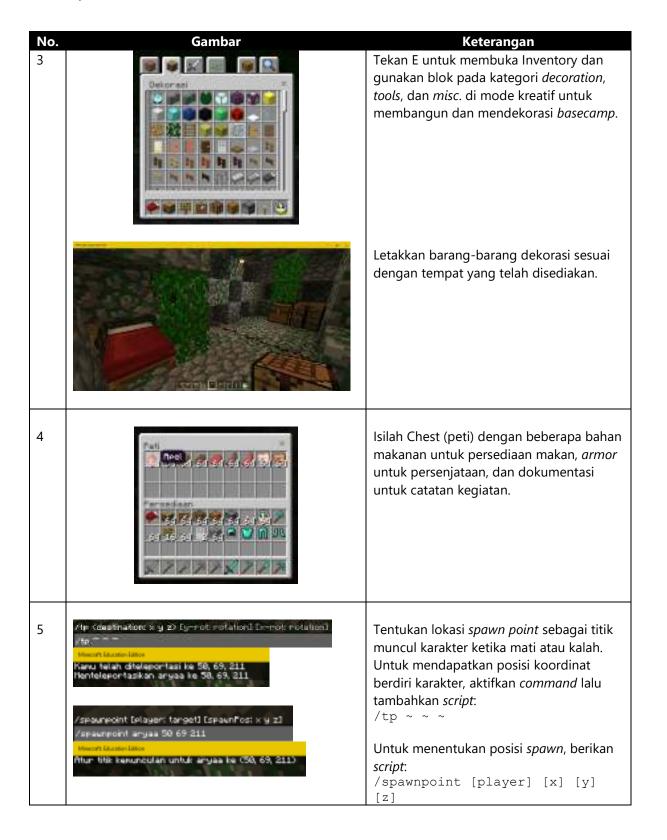
# 3.2.1 Langkah ke 1-Menentukan Basecamp

Bagian ini menceritakan langkah awal di mana Alex sebagai Ranger harus menentukan lokasi basecamp untuk melaporkan keadaan hutan lindung kepada Kesatuan Pusat Ranger di kota. Permainan akan dimainkan dalam mode kreatif untuk membangun basecamp, menambahkan dekorasi di basecamp dan area hutan lindung sebelum game dimainkan dalam survival mode.

# Objektif:

- 1. Membangun basecamp, menambahkan dekorasi basecamp.
- 2. Mengisi peti dengan makanan, armor, dan alat bantu.
- 3. Menentukan lokasi spawn point.



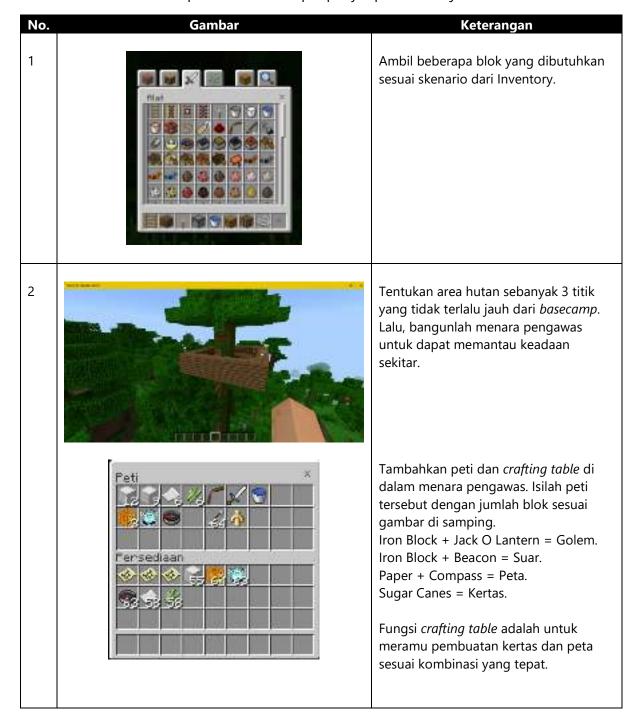


# 3.2.2 Langkah ke 2-Eksplorasi Hutan

Bagian ini menceritakan mengenai eksplorasi hutan yang nantinya akan dilakukan oleh Alex di tiga titik. Alex harus membangun menara pengawas sebagai tempat pemantau keadaan hutan dan dispenser sebagai tempat penyimpanan barang atau air ketika nanti terjadi kebakaran. Dispenser juga dapat digunakan sebagai tempat untuk menaruh jebakan seperti panah dan api.

# Objektif:

- 1. Membangun menara pengawas.
- 2. Menambahkan dispenser untuk tempat penyimpanan atau jebakan.





# 3.2.3 Langkah ke 3-Menambahkan Hostile Mob

Bagian ini menceritakan mengenai menambahkan *hostile mob* sebagai musuh yang harus dihadapi oleh Alex nantinya. *Hostile mob* ini diibaratkan sebagai kontraktor yang berencana melakukan penebangan liar dan kebakaran hutan. Mereka terdiri dari Vindicator (penyihir), Evoker (penebang), dan Ghast (penembak api). Hostile mob ini dapat dibuat dengan menggunakan *spawn egg* yang ada di Inventory bagian kategori Tools.

# Objektif:

1. Mengalahkan hostile mob.





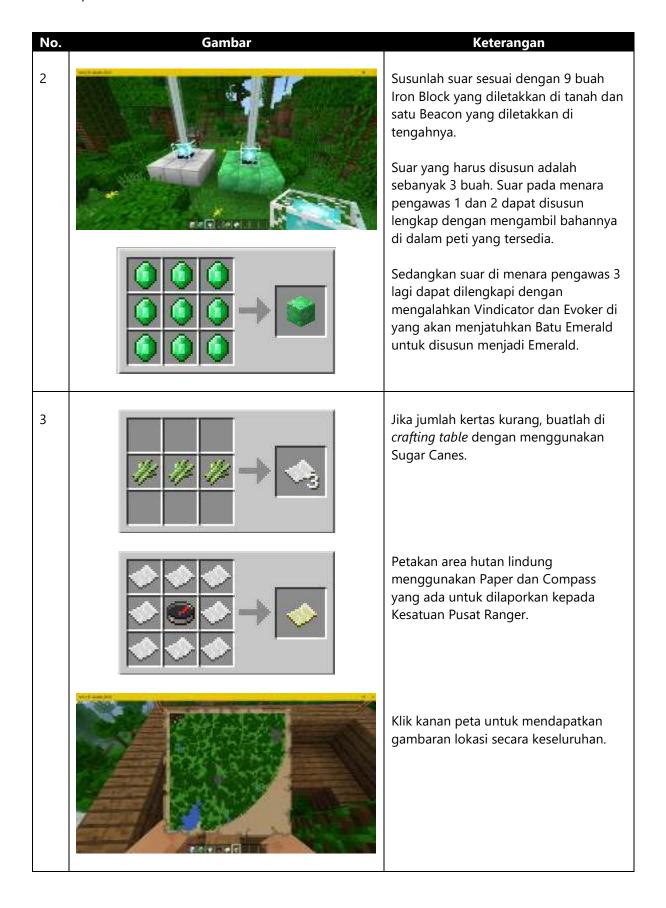
# 3.2.4 Langkah ke 4-Membuat Suar dan Memetakan Lokasi Hutan

Bagian ini menceritakan bagaimana Alex harus membuat suar untuk menandai lokasi area hutan lindung yang hampir saja dirusak oleh kontraktor sebanyak 3 buah. Lalu, Alex juga harus memetakan lokasi hutan dengan Paper dan Compass yang ada. Suar dapat dibangun menggunakan Beacon dan Batu Emerald.

# Objektif:

- 1. Membuat suar sebagai penanda lokasi.
- 2. Memetakan lokasi hutan lindung.



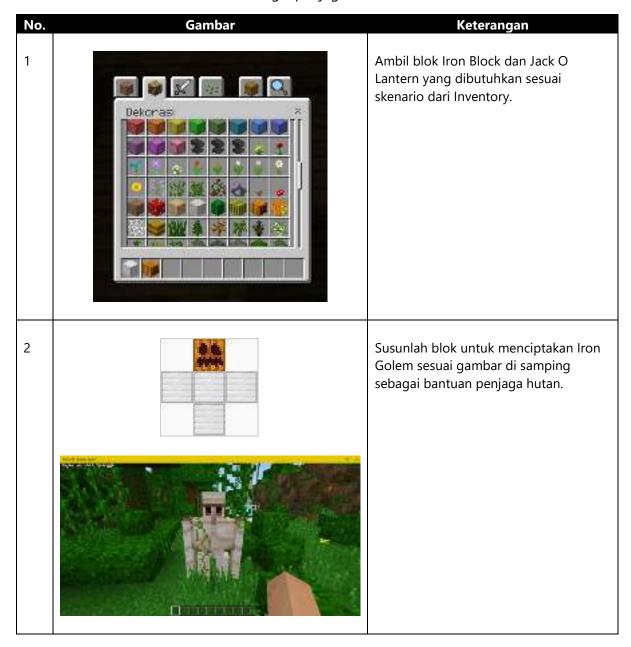


# 3.2.5 Langkah ke 5-Menambahkan Penjaga Hutan

Bagian ini menceritakan bagaimana Alex menyusun blok untuk menciptakan Iron Golem sebagai penjaga yang akan membantu area hutan lindung yang sebelumnya telah diserang oleh kontraktor.

# Objektif:

1. Menambahkan Iron Golem sebagai penjaga hutan.

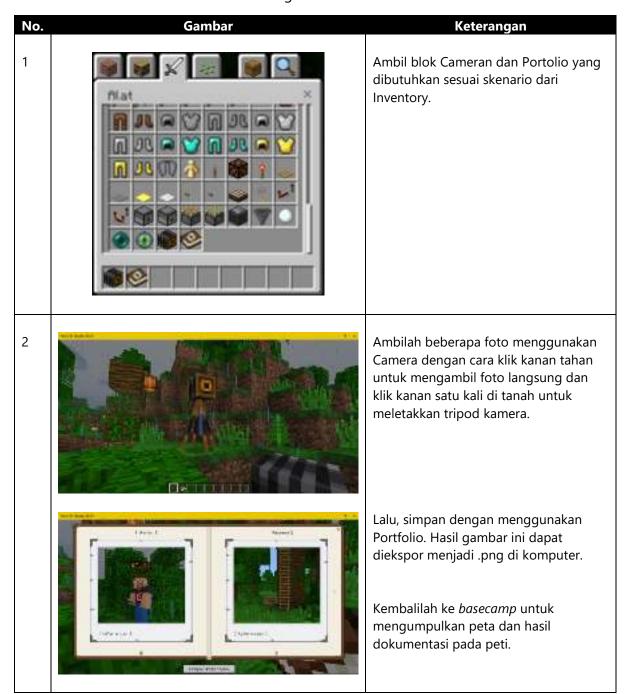


# 3.2.6 Langkah ke 6-Dokumentasi Hutan Lindung

Bagian ini menceritakan mengenai dokumentasi hutan lindung yang telah diselamatkan oleh Alex dari kontraktor dengan menggunakan Camera dan Portfolio sebagai bukti yang akan dikirimkan kepada Kesatuan Pusat Ranger.

# Objektif:

1. Mendokumentasikan hutan lindung.



# 3.2.7 Langkah ke 7-Mainkan dalam Mode Bertahan Hidup

Bagian ini menceritakan bagaimana mode permainan diganti menjadi mode survival Alex akan memiliki darah dan rasa lapar ketika bertugas menyusuri hutan dan harus menyelesaikan misi secepat mungkin. Ulangi mode bertahan hidup seperti mode kreatif yang baru saja dilakukan (langkah 2-6).

# Objektif:

1. Menyelamatkan Hutang Lindung Minecraftopia.



Gambar 2. 3 Alex Ranger Minecraftopia akan Dimainkan dalam Mode Bertahan Hidup

## Referensi

- AB, M. (2017, June 08). *Minecraft Wiki*. Diambil kembali dari Gamepedia: http://minecraft.gamepedia.com/Minecraft
- Booker, L. (2008, July 14). *Pandemic Working On New 'Open World / Sandbox' IP*. Diambil kembali dari Kotaku: https://www.kotaku.com.au/2008/07/pandemic\_working\_on\_new\_open\_world\_sandbox\_ip/
- Danny. (2015, November 30). 5 Top Tips For Games Writing. Diambil kembali dari Danny Stack Scripwriting in the UK: http://dannystack.com/5-top-tips-for-games-writing/
- Derk. (2012, August 06). *HOW TO WRITE A GOOD GAME STORY*. Diambil kembali dari Paladin: http://www.paladinstudios.com/2012/08/06/how-to-write-a-good-game-story-and-get-filthy-rich/
- Kantilaftis, H. (2016, April 01). *How To Write A Compelling Game Story In Three Steps*. Diambil kembali dari New York Film Academy Student Resources: https://www.nyfa.edu/student-resources/how-to-write-a-compelling-game-story-in-three-steps/
- Lee, T. (2013, November 04). *Designing game narrative: How to create a great story*. Diambil kembali dari Develop: http://www.develop-online.net/opinions/designing-game-narrative-how-to-create-a-great-story/0185460
- Schell, J. (2008). The Art of Game Design A Book of Lenses. Massachusetts: Morgan Kaufmann.

# Modul 4 - GameCapstone

# Pengembangan Game Berbasis VPL



# Didukung oleh:



# 4 PENGEMBANGAN GAME BERBASIS VISUAL PROGRAMMING LANGUAGE

Seperti yang kita ketahui, pogram adalah sekumpulan instruksi yang digunakan untuk mengatur kerja komputer sehingga dapat melakukan suatu tindakan tertentu. Tanpa program, komputer tidak dapat melakukan operasinya. Suatu program ditulis dengan sebuah bahasa pemrograman. Namun pada dasarnya mesin komputer hanya bekerja dengan hanya mengenali angka 0 dan 1, atau disebut dengan bilangan biner. Sedangkan, manusia dapat mengerjakan suatu intruksi berdasarkan kalimat-kalimat. Jadi, perlu adanya jembatan penghubung agar komputer mengerti instruksi yang diberikan oleh manusia, yaitu bahasa pemrograman.

Dalam komputasi, bahasa pemrograman visual atau *visual programming language* (VPL) adalah varian bahasa komputer yang memungkinkan pengguna membuat program dengan memanipulasi elemen program dalam bentuk grafis, bukan dengan teks (Jost, Ketterl, Budde, & Leimbach, 2014). VPL memungkinkan pemrograman dengan ekspresi visual dan simbol grafis. Sebagai contoh, banyak VPL (dikenal sebagai *dataflow* atau *diagram programming*) berdasarkan tampilan "kotak dan panah", di mana kotak atau objek layar lainnya berperan sebagai entitas, lalu dihubungkan oleh anak panah, garis atau busur yang merepresentasikan hubungan (Dehouck, 2015).

# **Tujuan Modul**

- 1. Mengenali fitur-fitur utama dalam Kodu Game Lab.
- 2. Memahami keunggulan dan kekurangan Kodu Game Lab dalam pengembangan permainan.
- 3. Mampu membuat aplikasi permainan dengan Kodu Game Lab.

# 4.1 Ringkasan Pembelajaran

Modul ini akan membahas mengenai pengembangan game berbasis *Visual Programming Language* (VPL) dengan menggunakan aplikasi Kodu Game Lab. Kodu merupakan sebuah aplikasi pembuat permainan digital *(game)* yang menggunakan bahasa pemrogaman visual untuk menyusunan kerangka pernainan dan logika di dalamnya. Dalam dunia pendidikan, selain mengasah kreativitas, Kodu juga dapat dioptimalkan untuk mengajarkan kepada pada siswa berpikir sistematis dan logis. Kodu didistribuskan secara gratis dan dilengkapi dengan visual grafis 3D yang menyenangkan. Kodu diciptakan dengan antarmuka pemrograman berbahasa sederhana dan sepenuhnya berbasis ikon.

Selanjutnya, setelah mengenal dasa-dasar Kodu, pengguna akan membuat sebuah game berjudul Momonga Pinball Adventures dengan menggunakan Kodu Game Lab.

# 4.1.1 Pengenalan Kodu Game Lab

Kodu (awalnya bernama Boku) adalah *programming integrated development environment* (IDE) yang dikembangkan oleh Microsoft FUSE Labs dan merupakan bahasa pemrograman visual yang dapat digunakan oleh siapa pun seperti proyek AgentSheets dan Squeak and Alice. Kodu Game Lab dapat berjalan di Xbox 360, Microsoft Windows XP, Windows Vista, Windows 7, Windows 8 dan Windows 10 (Labs, 2009).



Gambar 1. 1 World Design Kodu Game Lab

Namun ada beberapa perbedaan yang terdapat dalam Kodu Game Lab dengan yang lainnya, Kodu menghindari penulisan pengetikan kode dengan langsung membuat pengguna dapat membuat program melalui ikon elemen visual dan tampilan program dijalankan dalam lingkungan simulasi 3D. Kodu Game Lab telah digunakan sebagai alat pembelajaran pendidikan di beberapa sekolah dan pusat pembelajaran (Milson, 2015).

Model pemrograman Kodu sangat sederhana dan bisa diprogram menggunakan *game controller* atau kombinasi antara *keyboard* dan *mouse*. Bahkan fitur pemrograman di dalamnya terdapat beberapa bagian pemrograman seperti variabel simbolis, percabangan, perulangan, manipulasi jumlah dan *string*, subrutin, polimorfisme, dan sebagainya. Terdapat dua kondisi pemrograman dalam Kodu, yaitu WHEN dan DO. Singkatnya, <condition> <a condition> <

Misal, untuk menggerakkan karakter pengguna hanya tinggal memberikan kode:

<condition> <action>

WHEN keyboard arrows DO move quickly



Gambar 1. 2 Contoh Pemrograman Kode Kodu Game Lab

Sebelum memulai merancang permainan dengan Kodu, berikut beberapa hal yang perlu diperhatikan seputar aplikasi Kodu Game Lab dan ide permainan yang dapat diimplementasikan mengingat bahwa Kodu Game Lab juga memiliki keterbatasan.

Tabel 1. 1 Keterangan Aplikasi Kodu Game Lab

Kategori	Bisa	Tidak Bisa	
Instalasi Kodu	<ul> <li>Kodu adalah aplikasi GRATIS yang dapat dijalankan di komputer bersistem operasi Windows (7, 8, 8.1, atau 10) dengan spesifikasi sebagai berikut:         <ul> <li>A graphics card that supports DirectX 9.0c and Shader Model 2.0 or higher is requiredNET Framework 4.0 or higher is required.</li> <li>2 GB RAM, 4 GB recommended</li> <li>3D accelerated card with 128 megabytes (MB) RAM; discrete graphics card with dedicated 256-MB VRAM recommended for higher performance</li> <li>XGA (1024 x 768) or higher resolution monitor</li> </ul> </li> </ul>	Kodu dipasang di komputer dengan spesifikasi komputer yang lebih rendah.	
Penggunaan	Aplikasi dapat digunakan di komputer dengan <i>mouse</i>	Menggunakan Kodu di	
Aplikasi	dan <i>keyboard</i> . Untuk pengguna Windows 8 dan	komputer tanpa <i>mouse</i>	
	Windows 10 juga dapat mengunduh aplikasi dari Store	dan <i>keyboard</i> akan	
	dan meggunakan layar sentuh.	menyulitkan.	
Online atau	Aplikasi Kodu berjalan offline di komputer. Tidak Melakukan kolaborasi		
Offline	membutuhkan konektivitas internet saat memainkan. permainan.		
Aplikasi Tidak	Ketika melakukan instalasi Kodu terdapat 2 shortcut Menghapus aplik		
Terbuka	yang diberikan, selain aplikasinya ada Configure Kodu		
Sempurna	Game Labs. Atur grafik sesuai dengan kapabilitas		

Kategori	Bisa	Tidak Bisa
	komputer. Komputer dalam keadaan online, sehingga	
	sistem melakukan pengecekan pembaruan.	
Objek di Dalam	Pengguna hanya bisa menggunakan objek yang ada di	Pengguna tidak bisa
Kodu	dalam aplikasi.	memasukkan objek
		lain, untuk karakter dll
		dari luar aplikasi.
Simpan Data	Game Kodu dapat disimpan dengan ekstensi *.Kodu2,	Game Kodu tidak
	dan dapat dimainkan di komputer yang juga memiliki	dapat dikonversi ke
	aplikasi Kodu.	platform lain seperti
		Android misalnya.

Tabel 1. 2 Keterangan Ide Permainan Kodu Game Lab

Kategori	Bisa	Tidak Bisa
Jenis Permainan	Kodu dapat digunakan untuk mendesain berbagai	Mendesain
	jenis mainan, di antaranya shooting, racing,	permainan di luar
	collecting, maze/puzzle, adventure , role-playing.	jenis tersebut.
Permainan Tembak	Beberapa objek di Kodu dapat diprogram untuk	Membuat game
(Shooting)	mampu menembak ketika diberikan perintah	tembak FPS
	ataupun secara otomatis. Objek yang ditembak	seperti Call of
	dapat dengan otomatis hancur. Setiap kali objek	Duty atau Counter
	berhasil ditembak dapat diprogram untuk	Strike.
	penambahan nilai.	
Permainan Balapan	Kodu memiliki karakter Cycle yang dapat didesain	-
(Racing)	memiliki akselerasi laju yang cepat. Ground/lahan	
	bermain di Kodu dapat didesain mirip dengan	
	arena balap. Pengguna dapat membuat objek	
	untuk menentukan garis finish dan start.	
	Pengguna dapat mengatur gerak objek secara	
	otomatis dengan jalur yang sudah ditentukan.	
Permainan Mengoleksi	Ground/lahan bermain di Kodu dapat didesain	-
Sesuatu	sekompleks mungkin untuk membuat objek	
	tertentu tersembunyi. Karakter yang digunakan	
	untuk bermain dapat diprogram untuk dapat	
	menabrak atau menembak objek tertentu untuk	
	menghasilkan poin.	
Maze/Puzzle	Membuat game dengan tipe labirin, seperti game	Membuat game
	ala PacMan, Lode Runner atau Bomberman.	sepert Jigsaw
		Puzzles, Candy
		Crush.
Adventure	Terdapat berbagai jenis karakter dengan	-
	karakternya masing-masing, ada karakter yang	
	bisa terbang, bisa berenang, bisa menembak dan	

Kategori	Bisa	Tidak Bisa
	sebagainya. Dapat mendesain area bermain dalam	
	bentuk pegunungan, danau, dataran rendah atau	
	mengganti jenis tanah	
Role-Playing Game	Pengguna dapat mengatur komponen	-
	logika/program di setiap objek. Laju karakter	
	dapat ditentukan dengan alur yang telah	
	didefinisikan	
Karakter Kodu	Karakter Kodu adalah makhluk luar angkasa. Tidak	Membuat game
	memiliki tangan yang bisa digerakkan untuk	fighting
	memukul atau menendang.	
Membuat Game	Game di Kodu dapat dibuat lebih dari satu level.	-
Berlevel		
Penilaian	Game di Kodu dapat menerapkan sistem skor.	-
	Setiap kali pencapaian dapat diberikan skor.	
Game Over	Dengan waktu tertentu, atau gagalnya misi	-
	tertentu (tertembak, kalah cepat saat balapan dll)	
	bisa diprorgam untuk game over.	

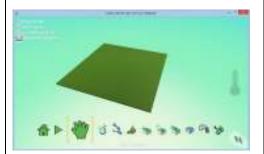
# 4.1.2 Fitur Game Development dalam Kodu Game Lab

Berikut beberapa fitur editor pada World Design Kodu Game Lab yang dapat digunakan untuk membuat permainan





#### **Move Camera**



Menggerakkan kamera

- 1. Bagi pengguna perangkat layar sentuh, dapat langsung mengelola/menggeser dengan kapabilitas *touch* yang dimiliki.
- 2. Bagi pengguna layar biasa gunakan klik kiri (ditahan) untuk menggeser, klik kanan (ditahan) untuk memutar, *scroll* untuk melakukan *zoom-in* dan *zoom-out*.

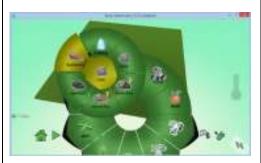
Proses ini digunakan untuk memudahkan pengguna dalam membuat dunia baru.



## **Objects Tools**



Tampilan komponen utama



Tampilan detail dari komponen utama

Berfungsi membuat berbagai komponen dalam dunia Kodu.

#### Penggunaan:

- Pilih menu Object Tools dengan klik kiri pada Object Tools.
- 2. Klik kiri pada dunia untuk memunculkan berbagai pilihan objek yang akan ditambahkan.

#### Contoh:

- 1. Klik kiri menu Object Tools.
- 2. Klik kiri pada dunia (menampilkan berbagai pilihan objek).
- 3. Klik kiri Rock (menampilkan pilihan tambahan sesuai dengan objek yang dipilih sebelumnya).
- 4. Klik kiri Sedimental (batu Sedimental akan tampil pada dunia di tempat kita meng-klik kiri di dunia sebelumnya).

Menu pengelolaan objek hanya akan aktif ketika ikon Object Tools dipilih, dan ketika dipindahkan ke menu lain, maka pengaturan objek tidak dapat dilakukan, termasuk untuk memindahkan objek.



Setiap objek juga dapat diubah warnanya sesuai dengan keinginan. Untuk mengubah warna sebuah objek:

 Pilih objek yang ada dalam dunia (dengan mengarahkan kursor pada objek) maka akan muncul tampilan pemilihan warna di bagian atas.

## Memilih warna untuk objek



Tampilan menu pada sebuah objek

2. Pilih warna yang diinginkan dengan menggunakan tombol arah kiri/kanan untuk memilih warna.

Pengaturan tambahan dapat ditampilkan dengan cara:

- 1. Tunjuk objek yang akan kita berikan pengaturan tambahan
- 2. Klik kanan objek tersebut

Beberapa pilihan menu yang tersedia, yaitu:

- 1. **Program**; Digunakan untuk memasukan logika/aksi dari objek tersebut (biasanya dilakukan untuk objek hidup).
- Change Setting; Untuk mengganti setelah objek, dapat digunakan untuk ketika membuat game dengan kompleksitas tinggi.
- 3. **Cut**; Untuk memindahkan objek ke halaman lain.
- 4. **Copy**; Untuk menyalin objek.
- 5. **Change Size**; Untuk mengatur ukuran objek.
- 6. Rotate; Untuk memutar tata letak objek.
- 7. **Change Height**; Untuk mengatur tinggi objek.

# A.

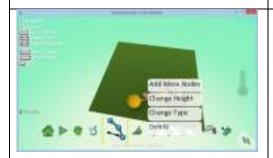
## **Path Tools**



Pembuatan Path

# Berfungsi untuk:

- 1. Mengatur peletakan objek, baik benda mati ataupun bergerak.
- 2. Menggerakkan objek ke arah tertentu pada dunia yang telah dibuat.
- 3. Mengatur arah jembatan, dinding, deretan tumbuhan dan sebagainya.



Tampilan menu pengaturan Path

Setiap Node (titik dalam Path) dapat dibuat dan diarahkan ke segala penjuru di dunia Kodu. Klik kanan untuk menampilkan pengaturan tambahan pada Path, menu tambahan pada Path:

- Add More Node; Untuk menambahkan Node baru (bisa dilakukan dengan langsung klik kiri pada dunia Kodu).
- Change Height; Untuk mengatur tinggi Path.
- Changet Type; Untuk mengatur tipe Path.
   Misalnya ketika dalam dunia memiliki berbagai objek, maka untuk membuat Path di masing-

masing objek perpindahan dari objek satu ke yang lainnya dapat menggunakan menu ini. **Delete**; Untuk menghapus Path (dapat dilakukan dengan cara yang lebih mudah, yaitu dengan menaruh kursor mouse di atas Node, kemudian tekan tombol Delete pada Keyboard). Berfungsi untuk memperluas dunia Kodu. Dunia Kodu dapat terdiri dari berbagai jenis tipe areal yang dapat dipilih pengguna disesuaikan dengan nilai estetika **Ground Brush** permainan. Operasi yang dapat dilakukan: Klik kiri (termasuk menekan lalu menggeser) digunakan untuk menambahkan area baru. 2. **Klik kanan** (termasuk menekan lalu menggeser) digunakan untuk menghapus objek yang telah dibuat. 3. **Tombola arah kiri/kanan** digunakan untuk memperkecil/memperbesar ukuran Ground Brush Terdapat banyak sekali tipe Ground yang dapat dipilih, Pemilihan tipe-tipe Ground terlebih dahulu pilih tipe Ground yang diinginkan. Cara memilih Ground: Menggunakan tombol arah kiri/kanan pada Menggunakan scroll pada mouse Tipe bentuk dapat dipilih untuk mempermudah pengguna dalam berkreasi memperluas cakupan dunia. Cara memilih Tipe Bentuk: Memilih bentuk objek penambahan Menggunakan tombol arah kiri/kanan pada Ground keyboard Menggunakan scroll pada mouse Berfungsi untuk membuat dataran tinggi dan dataran rendah. Klik kiri dan tahan untuk menambahkan dataran tinggi dan klik kanan dan tahan untuk **Hills and Valleys** menambahkan dataran rendah, termasuk untuk membuat yang tinggi menjadi rendah atau sebaliknya. Berfungsi untuk menghaluskan gradasi objek yang ditimbulkan dari pembuatan dataran tinggi ataupun

**Flatten** 

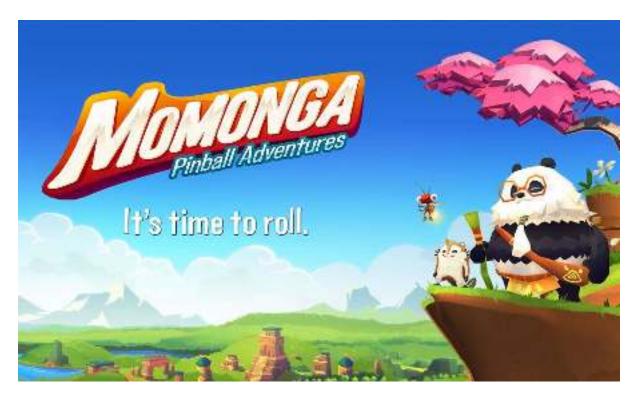
dataran rendah, sekaligus untuk memperjelas

perbedaan antara objek tinggi dan rendah.

Roughen	Berfungsi digunakan untuk membuat areal objek menjadi lebih tinggi. Cara penggunaan klik kiri/kanan pada mouse dan tahan untuk meninggikan tiap areal yang tersorot.
Water Tool	Berfungsi untuk menambahkan genangan air pada objek-objek datar yang ada. Cara penggunaan:  1. <b>Klik kiri</b> untuk menambahkan intensitas air 2. <b>Klik kanan</b> untuk mengurangi intensitas air
Delete Tool	Berfungsi untuk penghapusan objek secara mudah dan cepat.
World Setting	Berfungsi untuk pengelolaan dunia dan berbagai hal terkait dengan tampilan di layar.
Home	Berfungsi untuk mendapatkan berbagai perintah awal sistem, seperti untuk menyimpan dunia yang telah dibuat atau mencetak kode logika jika sudah ada program yang ditambahkan di objek.

# 4.1.3 Momonga Pinball Adventures

Momonga adalah *game* yang mengusung tema *pinball* dengan *gameplay* unik yang dikembangkan oleh Game Troopers. Game ini dapat dimainkan di Windows Phone dan Windows 10.



Gambar 1. 3 Momonga Pinball Adventures

Lalu, bagaimana storyline-nya? Terdapat sebuah benua bernama Aya dimana para hewan hidup damai sejak berakhirnya Perang Great War. Para Dewan Wali The Guardian Council dibentuk untuk melindungi Sources dan dunia dari perang. Sources adalah lima tempat dengan kekuatan besar yang membawa keseimbangan kepada dunia jika mereka berada di tangan yang tepat. Sejak dimulainya Perang Great War dahulu, Sources telah diperebutkan oleh Shadow Species yang menginginkan Sources untuk diri mereka sendiri.

Selanjutnya, momonga adalah suku kecil dari tupai terbang yang tinggal di sekelompok pulau kecil di dekat sungai Yana di mana mereka membuat rumah di pohon Sequoia raksasa. Suatu hari, desa tersebut diserang oleh sekelompok bandit burung hantu, The Owl. Pemimpin mereka, Jenderal Kuton adalah seorang jenderal pemberontak di masa lalu yang kehilangan anaknya di Ravens. The Owl mengambil semua momonga kecuali satu momonga muda bernama Momo. Lalu, ia diselamatkan oleh Panda yang sedang berkeliaran dan diajak ke pegunungan untuk melatih Momo menjadi lebih kuat demi menyelamatkan sukunya.



Gambar 1. 4 Gamepley Momonga Pinball Adventures

Pemain dapat memainkan fitur permainan Momonga Pinball Adventures seperti mode pinball (memukul sasaran panah dan menuju checkpoint), mode flipper (terbang untuk mengumpulkan objek bintang), mode bossfight (melawan boss untuk melanjutkan perjalanan) sembari mengumpulkan bintang yang menjadi parameter keberhasilan player menyelesaikan setiap level.

Catatan: Tiga misi pertama gratis. Misi lainnya tersedia dalam In-App-Purchase.

https://www.microsoft.com/id-id/store/p/momonga-pinball-adventures/9nblggh20bpg

# 4.2 Video Tutorial URL

Bagian ini akan membahas studi kasus pembuatan game Momonga Pinball Adventures dengan menggunakan Kodu Game Lab. Game yang dikembangkan adalah Momodu (Momonga ala Kodu) dengan *gameplay* yang tidak akan sama persis 100% karena keterbatasan Kodu Game Lab. Berikut beberapa perbandingan antara game Momonga Pinball Adventures dan Momodu (Momonga ala Kodu).

Tautan Video Tutorial: <a href="https://youtu.be/t4f6H3y409k">https://youtu.be/t4f6H3y409k</a>.

Tabel 1. 3 Perbandingan Momonga dan Momodu

# Level **Momonga Pinball Adventures** Momodu (Momonga ala Kodu) Level 1 1. Kontrol *flipper* pinball menggunakan 1. Kontrol Boku menggunakan tombol tombol kiri dan kanan dan space untuk arah keyboard, A dan D untuk mengambil dan menjatuhkan bola. melepaskan Momo. 2. Goal menaruh bola di atas tanah hitam 2. Goal menghancurkan papan sasaran dan menuju ke arah lingkaran biru. untuk mendapatkan bintang. 3. Item bintang dapat diambil sebagai Game over jika darah habis karena reward jika memenuhi target. bertabrakan dengan batu dan akan 4. Game over jika darah habis dan jatuh mental jika bertabrakan dengan pipa. ke lubang hitam. Level 2 1. Kontrol terbang Momo menggunakan Kontrol Jet menggunakan tombol arah tombol kiri dan kanan untuk terbang. keyboard untuk bergerak, W dan S 2. Goal menghancurkan papan sasaran untuk menaikkan dan menurunkan jet. dan menuju ke arah lingkaran biru. 2. Goal mengambil koin sampai mencapai 3. Item bintang dapat diambil sebagai 100 poin, lalu masuk ke dalam awan reward jika memenuhi target. merah sebelum timer habis. Game over jika darah habis dan jatuh 3. Game over jika darah habis karena ke lubang hitam. terkenan lemparan batu dan timer menjadi 0 detik.

Level	Momonga Pinball Adventures	Momodu (Momonga ala Kodu)
Level 3		10 to 10
	Kontrol <i>flipper</i> pinball menggunakan tombol kiri dan kanan untuk terbang.	Kontrol Cycle menggunakan tombol arah kiri dan kanan untuk bergerak, Q
	Goal mengalahkan Panda dengan memukul tongkat bambu.	untuk menembakkan misil.  2. <i>Goal</i> menembakkan misil ke kastil
	3. Item bintang dapat diambil sebagai reward jika memenuhi target.	hitam hingga mencapai skor 20 poin.  3. <i>Game over</i> tidak ada karena harus
	4. <i>Game over</i> jika darah habis dan jatuh ke lubang hitam.	berhasil.

# 4.2.1 Level 1-Momodu (Pinball Mode)

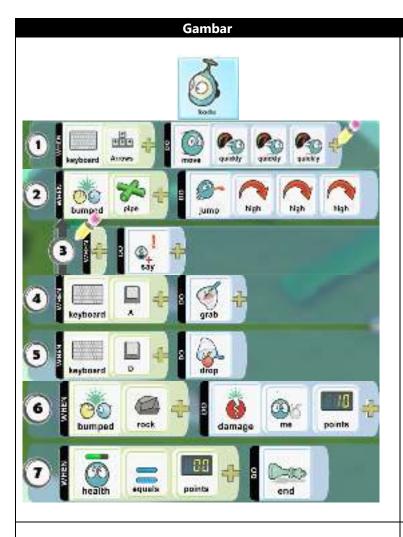
Di dalam Kodu Game Lab, objek yang mirip dengan *flipper* pinball pada Momonga adalah pipa. Namun, pengguna tidak bisa membuat objek seperti pipa dan benda mati lainnya untuk diberikan program kontrol bergerak. Lalu, bagaimana membuat Momodu dengan Pinball Mode seperti Momonga? *Gameplay* di level ini adalah menggerakkan karakter robot bernama Boku untuk mengambil bola sepak dan menaruhnya di atas tanah hitam. Jika bola sepak berhasil diletakkan akan muncul bintang yang akan membimbing Boku ke level selanjutnya. Boku juga akan digerakkan menyusuri arena pin ball yang tersusun dari pipa dan berbatuan.

Gambar

Keterangan

Langkah 1-Membuat Terrain
Buatlah arena pinball dengan
membuat area terrain yang cukup
besar menggunakan Ground
Brush.

Tabel 1. 4 Langkah Pengembangan Level 1 Momodu



#### \_ \_ \_ \_

Keterangan

#### Langkah 2-Karakter Boku

Tambahkan karakter Boku. Lalu berikan program berikut.

Baris 1: Tombol arah keyboard untuk menggerakkan karakter.

Baris 2 & 3: Jika bertabrakan dengan pipa akan mental dan mengatakan OUCH!.

Baris 4 & 5: Tombol A dan D untuk mengambil dan menjatuhkan bola sepak

Baris 6: Jika bertabrakan dengan batu, darah akan berkurang.

Baris 7: Game over jika darah habis.



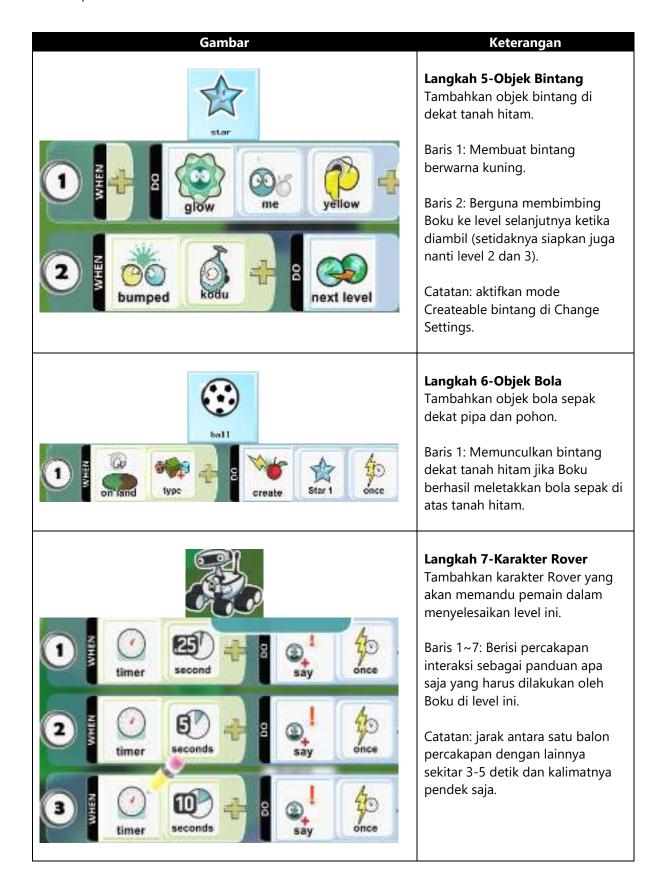
# Langkah 3-Desain Dunia

Tambahkan objek pipa, berbatuan dan pohon sesuai dengan gambar di samping untuk membuat dunianya persis seperti *pinball*.



### Langkah 4-Membuat Tanah Hitam

Tambahkan tanah hitam dengan mengganti jenis Groud Brush yang ada sebagai *goal* level 1 ini.



# 4.2.2 Level 2-Momodu (Fly Mode)

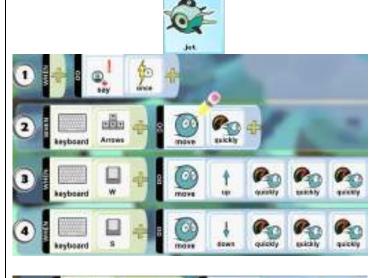
Level ini hampir mirip dengan *fly mode* di Momonga karena *gameplay* yang dimainkan cukup mudah dengan hanya mengarahkan arah terbang Momo sembari mengumpulkan bintang. Pada Momodu yang akan dikembangkan, karakter Boku akan diganti dengan Jet. Misinya adalah mengambil seluruh koin yang tersebar di seluruh arena hingga skor menjadi 100 poin.

Tabel 1. 5 Langkah Pengembangan Level 2 Momodu



#### Keterangan

Langkah 1-Membuat *Terrain*Buatlah arena terbang untuk Jet
menggunakan Ground Brush yang
Circle. Lalu, tambahkan air di
atasnya.



## Langkah 2-Karakter Jet

Tambahkan karakter Jet. Lalu berikan program berikut.

Baris 1: Panduan memulai permainan yang ditampilkan secara *full screen*.

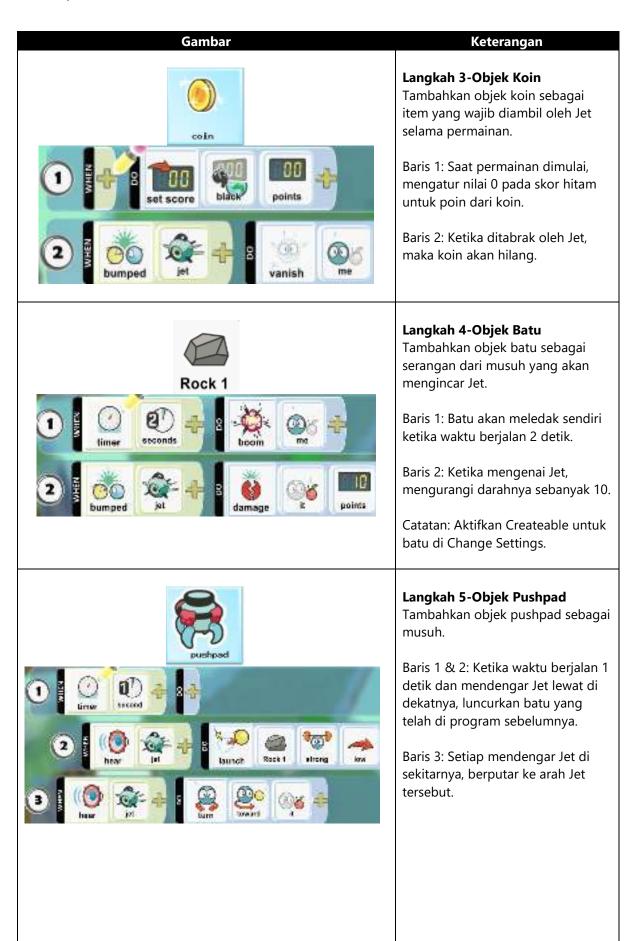
Baris 2: Tombol arah keyboard untuk menggerakkan Jet.

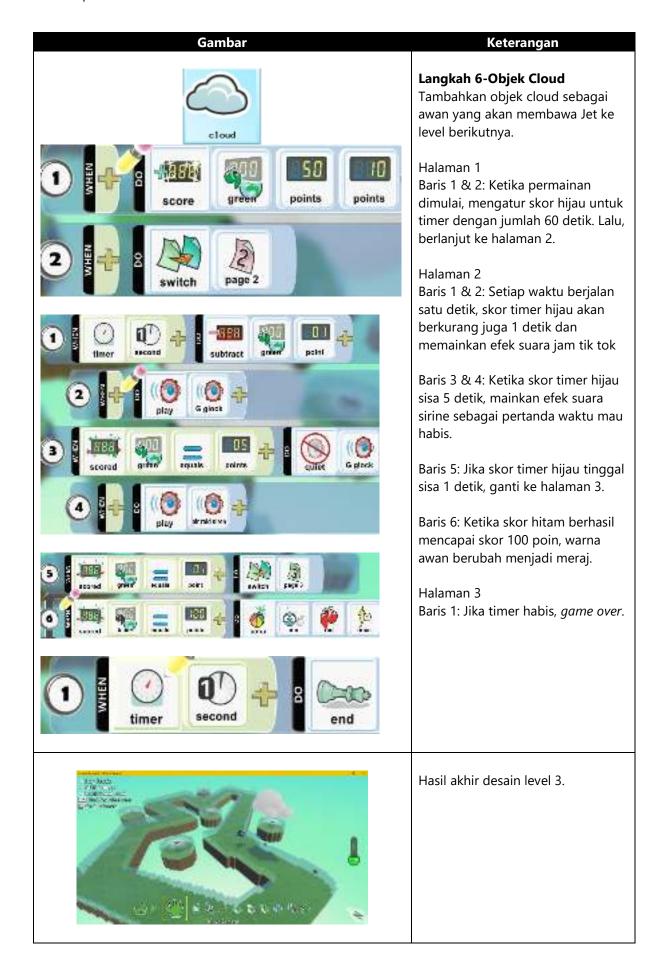
Baris 3 & 4: Tombol W dan S untuk menaikkan dan menurunkan Jet.



Baris 5: Setiap menabrak koin akan mendapatkan skor dengan warna hitam 5 poin.

Baris 6: Ketika Jet menabrak awan berwarna merah maka akan lanjut ke level berikutnya.





# 4.2.3 Level 3-Momodu (Challenge Mode)

Level ini adalah level terakhir. Karakter Jet akan diganti dengan Cycle. Misinya adalah menyelesaikan tantangan yang sama seperti di Momonga ketika Momo harus memukul tongkat bambu Panda. Bedanya, tongkat bambu diganti dengan bentuk kastil yang akan terus berganti warna setiap detik. Cycle harus menembakkan misil tepat ke kastil berwarna hitam untuk mendapatkan 20 poin dan menang.

Gambar

Tabel 1. 6 Langkah Pengembangan Level 3 Momodu

# Langkah 1-Membuat Terrain

Keterangan

Buatlah arena bermain berbentuk kotak menggunakan Ground Brush dengan tambahan jalur warna putih di sekeliling terrain menggunakan Path.







#### Langkah 2-Karakter Cycle

Tambahkan karakter Cycle. Lalu berikan program berikut.

Baris 1: Panduan memulai permainan yang ditampilkan secara full screen.

Baris 2 & 3: Tombol kiri dan kanan untuk menggerakkan arah Cycle.

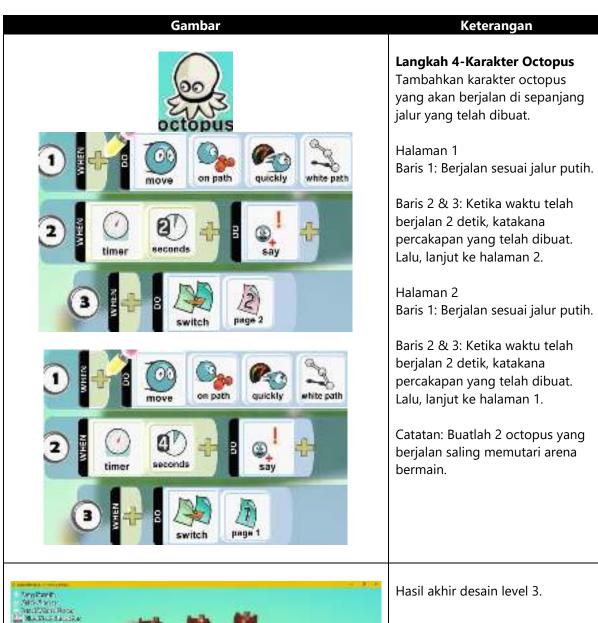
Baris 4: Tombol O untuk menembakkan misil.

Baris 5: Ketika permainan dimulai, mengatur skor warna hitam untuk poin dengan nilai 0.

Baris 6: Ketika misil berhasil mengenai kastil hitam, skor hitam bertambah 5 poin.

Baris 7: Jika skor hitam bernilai 20 poin maka permainan berhasil dimenangkan.





# Referensi

- Azure. (2017, April 24). *What is Azure?* Diambil kembali dari Azure: https://azure.microsoft.com/en-us/overview/what-is-azure/
- Dehouck, R. (2015, September 29). *The maturity of visual programming*. Diambil kembali dari Craft.Ai: http://www.craft.ai/blog/the-maturity-of-visual-programming/
- Jost, B., Ketterl, M., Budde, R., & Leimbach, T. (2014). Graphical Programming Environments for Educational Robots: Open Roberta Yet Another One? *2014 IEEE International Symposium on Multimedia*. Taichung, Taiwan: IEEE.
- Labs, M. R. (2009). *What is Kodu?* Diambil kembali dari Kodu Game Lab: https://www.kodugamelab.com/about/
- Milson, E. (2015, October 11). *Minecraft vs Project Spark vs Kodu Game Lab*. Diambil kembali dari MSDN: https://blogs.msdn.microsoft.com/austeachers/2015/10/11/minecraft-vs-project-spark-vs-kodu-game-lab/

UNIVERSITAS

# Modul 5 - GameCapstone

# Pengembangan Game Berbasis HTML5



Construct 2

# Didukung oleh:



# 5 PENGEMBANGAN GAME BERBASIS HTML5

HTML5 sendiri adalah sebuah bahasa markah untuk menstrukturkan dan menampilkan isi dari World Wide Web, sebuah teknologi inti dari Internet. HTML5 adalah revisi kelima dari HTML (yang pertama kali diciptakan pada tahun 1990 dan versi keempatnya HTML4, pada tahun 1997) dan hingga bulan Juni 2011 masih dalam pengembangan. Tujuan utama pengembangan HTML5 adalah untuk memperbaiki teknologi HTML agar mendukung teknologi multimedia terbaru, mudah dibaca oleh manusia dan juga mudah dimengerti oleh mesin (Group, 2011). Biasanya, *game browser* dikembangkan dengan menggunakan teknologi Flash dari Adobe. Namun, sekarang dapat juga dengan menggunakan HTML5. Salah satu perangkat lunak yang dapat dimanfaatkan untuk pembuatan *game* berbasis HTML5 adalah Construct 2.

# **Tujuan Modul**

- 1. Memahami karakteristik game browser.
- 2. Mengenali fitur-fitur utama dalam Construct2.
- 3. Memahami keunggulan dan kekurangan Construct2 dalam pengembangan permainan.
- 4. Mampu membuat aplikasi permainan dengan Construct2.

# 5.1 Ringkasan Pembelajaran

Game browser (permainan digital berbasis web) adalah platform yang sangat berkembang, khususnya di Indonesia. Game tersebut bisa dimainkan tanpa mengunduh dan instalasi client secara khusus. Namun, gamers tetap dituntut untuk mengunduh game assets atau data-data yang dibutuhkan untuk memainkan game tersebut dan menginstalasinya ke dalam cache pada browser yang digunakan untuk bermain (Fakry, 2012). Biasanya game browser dikembangkan dengan menggunakan Flash atau HTML5.

Sekarang pengguna bisa membuat *game* hanya dengan bermodalkan *text editor* dan *web browser* saja berkat perkembangan HTML5 dengan fitur *canvas* yang dapat dimanfaatkan untuk mengembangkan *game*. Salah satu tujuan pengembangan HTML5 adalah mengurangi ketergantungan akan aplikasi pihak ketiga seperti Flash dengan memberikan HTML fitur *native* untuk melakukan hal-hal yang dapat dilakukan oleh Flash, *canvas* merupakan salah satu fitur HTML5 untuk mengatasi permasalahan ini. *Canvas* sendiri digunakan oleh banyak aplikasi, mulai dari *audio player*, *video player*, *diagram responsive*, dan bahkan *game* (Bagus, 2015).

Salah satu perangkat lunak yang dapat dimanfaatkan untuk pembuatan *game* berbasis HTML5 adalah Construct 2. Modul ini akan membahas fitur-fitur utama dalam Construct2, memahami keunggulan dan kekurangan Construct2 dalam pengembangan permainan, dan mampu membuat aplikasi permainan dengan Construct2

# 5.1.1 Pengenalan Construct2

Construct 2 adalah *game creator* berbasis HTML5 yang dirancang khusus untuk *game* 2D dan dikembangkan oleh Scirra Ltd. Penggunaan kode pemrograman dalam Construct 2 menggunakan editor visual dengan cara *drag-and-drop* dan sistem logika berbasis *behavior* sehingga seseorang yang tidak mempunyai kemampuan dasar *coding* tetap dapat mengembangkan *game* menggunakan Construct 2 (Scirra, 2007). Construct 2 dapat diunduh melalui tautan di sini https://www.scirra.com/construct2/releases/new.



Gambar 5. 1 Tampilan Start Page Construct 2

Construct 2 digunakan dengan menggunakan lisensi. Tanpa lisensi, versi yang digunakan secara otomatis adalah Construct 2 Free Edition dengan keterbatasan hanya dapat menggunakan maksimal 100 events dalam sebuah project, maksimal 4 layers dalam layout, maksimal 2 effects dalam sebuah project, tidak ada fitur organisasi folder, tidak dapat mencari event, tidak ada konfigurasi bar, tidak diizinkan digunakan untuk tujuan komersial. Membeli lisensi akan menghapus semua batasan ini. Lisensi dapat dibeli dari Scirra.com.

Editor pada Construct 2 hanya dapat dijalankan di Windows. Namun, semua *game* HTML5 yang dikembangkan nantinya dapat diekspor pada berbagai perangkat. Berikut spesifikasi yang dibutuhkan untuk menjalankan Construct 2:

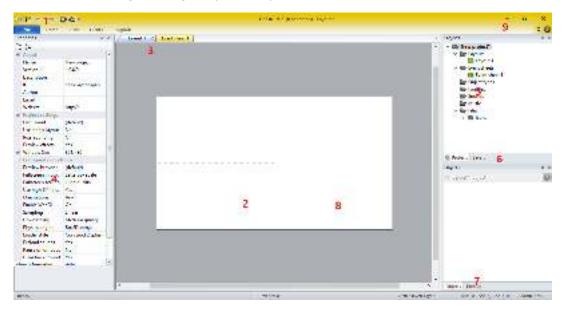
# Spesifikasi Minimum:

- Windows XP Service Pack 3 atau versi di atasnya.
- 512 MB RAM.
- 1 GHz Processor.
- Browser yang kompatibel dengan HTML5 (Internet Explorer 9+, Mozilla Firefox, Google Chrome, dan Opera).
- Graphics Card Drivers versi terbaru.

#### Spesifikasi yang Direkomendasikan:

- Windows 7 atau versi di atasnya.
- 2 GB RAM.
- 2 GHz dual-core processor.
- Graphics Card Drivers nVidia atau AMD.

Tampilan antarmuka Construct 2 mirip seperti aplikasi Office dan fungsi dari masing-masing fiturnya dapat dengan mudah dipelajari oleh pengguna. Berikut adalah tampilan antarmuka Construct 2 dan masing-masing penjelasannya (Gullen, 2011):



Gambar 5. 2 Tampilan Antarmuka Construct 2

#### 1. File Menu dan Tab Ribbon

Tombol biru dengan panah adalah menu File yang digunakan untuk membuat, membuka, menyimpan, mengekspor, dan menutup *project*. Ada juga menu Home, View, Events, dan Upgrade untuk mengatur editor, tampilan, *event*, dan fitur terbaru. Ada juga *toolbar* akses cepat dengan empat perintah yang paling umum digunakan: Save, Undo, Redo, Preview and Debug. Pengguna dapat menambahkan dan menghapus tombol di *toolbar* ini dengan klik panah *drop-down* kecil ke kanan untuk menyesuaikannya.

# 2. Tampilan Layout

Editor visual tempat menempatkan objek untuk elemen *game* seperti *background*, *sprite*, dan sebagainya.

#### 3. Tab

Tab ini memungkinkan Pengguna beralih di antara tampilan *layout* dan tampilan *sheet* yang berbeda untuk menentukan aktivitas logika permainan. Pengguna dapat menyeret tab ke salah satu tampilan untuk membuat tampilan layar berdampingan.

# 4. Property Bar

Berisi daftar properti tergantung pada apa yang telah Pengguna klik. Pengguna dapat melihat dan mengubah properti seperti tata letak, objek, lapisan dan lainnya.

#### 5. Tab Project Bar dan Layer Bar

Secara *default*, Project Bar dan Layers Bar berada di sini. Project Bar berisi gambaran umum tentang segala sesuatu dalam proyek Pengguna. Layer Bar digunakan untuk menambahkan lapisan yang berbeda ke tampilan tata letak.

#### 6. Tab Project Bar/Layer Bar

Memungkinkan Pengguna beralih antara Project Bar dan Layers Bar.

# 7. Object Bar

Berisi daftar objek dalam *game*. Pengguna dapat menarik dan melepaskan objek dari sini untuk menempatkannya dalam tata letak.

## 8. Status Bar

Di bagian bawah jendela editor terdapat Status Bar yang menampilkan informasi apa yang sedang dijalankan oleh Construct 2 saat menjalankan *project*, berapa banyak *events* dalam *project*, layer yang aktif, posisi koordinat mouse, dan tingkat *zoom*.

#### 9. Tombol Kanan Atas

Seperti halnya Windows, digunakan untuk meminimalkan, memaksimalkan dan menutup Construct 2.

# 5.1.2 Fitur Game Development pada Construct2

Bagi pengguna pemula yang ingin membuat *game* tapi tidak mengetahui konsep dasar pemrograman, Construct 2 adalah alat bantu yang sempurna. Pengguna dapat dengan cepat mempelajari tampilan antarmuka, *layout*, dan sistem *event* dalam membuat *game*. Berikut tabel penjelasan masing-masing fitur yang dapat dimanfaatkan ketika menggunakan Construct 2:

Tabel 5. 1 Fitur pada Construct 2



# Fitur Cepat & Mudah

Layout Editor menampilkan antarmuka visual yang dapat dengan cepat dan mudah dipahami oleh pengguna serta dapat menampilkan apa yang sedang dilihat adalah apa yang akan didapatkan selama proses perancangan *game* berjalan.

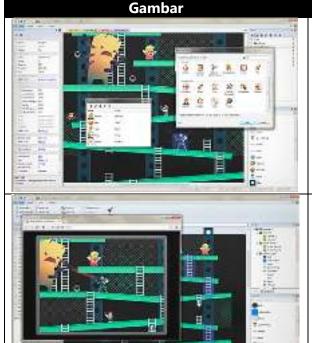
Pengguna dapat menarik, memutar dan mengubah ukuran objek, memvisualisasikan efek yang diterapkan, dan mengubah pengaturan sesuai dengan keinginan Pengguna di Property Bar.

Objek juga dapat diatur pada lapisan yang terpisah untuk kemudahan pengorganisasian berkas. Terdapat juga editor gambar *built-in* untuk memudahkan pengeditan cepat pada grafis objek Pengguna.



#### Sistem Event yang Powerful

Kode pemrograman yang ditulis ditampilkan dalam bentuk visual dan mudah dibaca sehingga memudahkan pengorganisasian daftar kode.





# **Behavior yang Fleksibel**

Behavior adalah perilaku yang dapat diterapkan pada objek sehingga mereka memiliki kemampuan seperti untuk arah ada 8 directions, platform, dan car; advance ada physics dan pathfinding; dan kemampuan seperti fade, flash, wrap, pin dan drag & drop.

# Pratinjau Instan

Construct 2 memungkinkan pengguna dapat melihat langsung pratinjau *game* yang sedang dikembangkan. Tidak perlu menunggu proses kompilasi yang memakan waktu. Pembuatan *game* menjadi lebih mudah bagi pemula. Fitur pengujian mudah ini adalah Preview Over Wifi sehingga *game testing* bisa dilihat pada ponsel, tablet, laptop atau PC lain yang terhubung di LAN / Wifi.



# **Efek Visual**

Terdapat lebih dari 70 efek pixel shader berbasis WebGL untuk warp, distort, blend, blur, mask, re-color dan masih banyak lagi. Pengguna dapat menambahkan efek visual pada objek, layer dan layout.

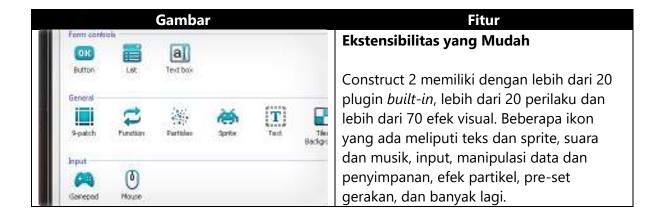
Fitur hebat lainnya adalah Particle plugin yang memungkinkan gerakan dari banyak gambar kecil secara seperti semprotan kilauan, asap, air, dan puing.



# **Ekspor Multiplatform**

Pengguna dapat mempublikasikan permainan di berbagai pilihan platform seperti di situs web pribadi, Chrome Web Store, Facebook, Kongregate, NewGrounds, Firefox Marketplace atau Scirra Arcade untuk membagikan *game*.

Penguna juga dapat mengekspor *game* ke PC desktop, Mac dan Linux dengan menggunakan Node-Webkit.



Tapi dengan berbagai macam kelebihan fitur yang ditawarkan, Construct 2 juga memiliki kekurangan. Berikut kelebihan dan kekurangan dari Construct 2 (Sidhion, 2012):

#### Kelebihan:

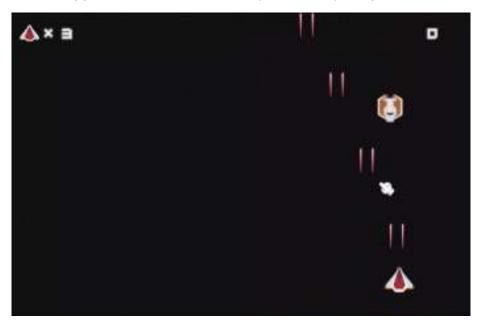
- Pengguna dapat membuat beberapa jenis permainan, seperti *platformer* dan *shooter* secara mudah dan cepat.
- Sistem behavior sangat intuitif dan mudah digunakan.
- Pengguna dapat membuat plugin khusus.
- Game yang dikembangkan dapat diekspor ke berbagai platform dan perangkat.

# Kekurangan:

- Tidak cukup deskriptif.
- Harus membaca banyak tutorial dan panduan sehingga pengembangan *game* dapat berjalan dengan baik.
- Tidak memiliki banyak fitur untuk penentuan posisi dan elemen visual di layar.
- Ekspor hanya sebagai game HTML5.

# 5.2 Video Tutorial URL

Game yang akan dikembangkan dengan menggunakan Construct 2 adalah Space War dengan tema *third person shooter. Gameplay* yang akan dimainkan adalah pengguna akan memainkan karakter pesawat luar angkasa dengan menembakkan laser ke pesawat musuh sampai mencapai skor tertinggi. Pemain akan diberikan nyawa sebanyak tiga buah.



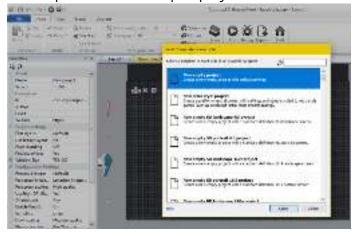
Gambar 5. 3 Gameplay Space War

Tautan Video Tutorial: <a href="https://youtu.be/POrXWshUMtQ">https://youtu.be/POrXWshUMtQ</a>.

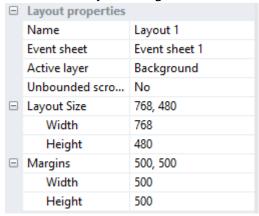
## 5.2.1 Langkah 1-Membuat New Project dan Mengatur Layout

Setiap *game* yang akan dikembangkan dari *scratch* dengan Construct 2 biasanya menggunakan **New Empty Project**. Lalu, pengguna wajib mengatur *layout* sehingga tampilan permainan ketika dijalankan dapat ditampilkan secara maksimal.

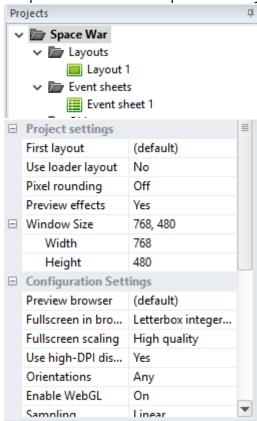
1. Buka Construct 2. Klik menu biru File, pilih New, lalu pilih New Empty Project. Berikan nama Space War dan simpan project.



2. Atur ukuran layout dengan ukuran 768 x 480.



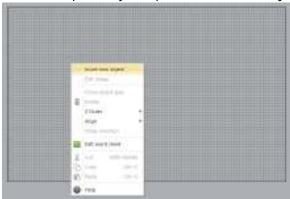
3. Klik nama project pada panel project di samping kanan dan ubah properti dari Windows Size dan Fullscreen in Browser sehingga tampilan layout sama dengan tampilan Windows ketika permainan dijalankan.



# 5.2.2 Langkah 2-Menambahkan Object

Object adalah aset gambar yang akan digunakan sebagai elemen *game* seperti background, player, enemy, laser, particle, font.

1. Klik kanan pada layout, pilih Insert New Object.



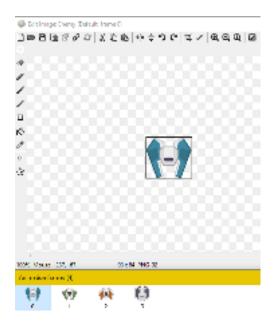
2. Masukkan aset gambar yang telah disiapkan seperti Background, Player, Enemy, Laser.



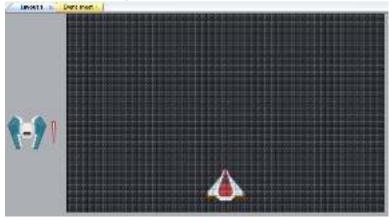
3. Tampilkan grid pada menu View dan ubah ukuran tinggi dan lebarnya untuk memudahkan meletakkan Background.



4. Tambahkan Enemy sebanyak 4 frame sehingga objeknya bisa dipanggil secara random nantinya.



5. Letakkan objek Player di dalam Background, sedangkan Enemy dan Laser di luar Background sehingga Enemy dan Laser tidak langsung muncul di dalam layout ketika nanti permainan dijalankan.



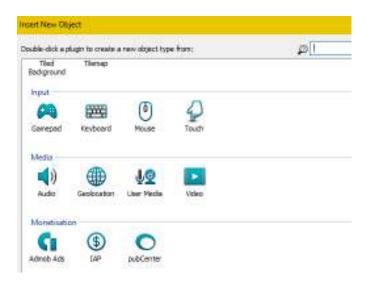
6. Klik kanan pada objek Player dan pilih Behavior. Tambahkan BoundToLayout untuk membuat gambar Player tidak bisa melewati batas Layout.



# 5.2.3 Langkah 3-Memberikan Kontrol Gerakan Player

Kode pemrograman pada Construct 2 dilakukan pada tab Event Sheet. Pertama, kontrol gerakan pesawat, yaitu Player akan digerakkan oleh Mouse. Lalu, akan diganti dengan Keyboard.

1. Tambahkan Insert New Object dan pilih Input untuk Mouse dan Keyboard.



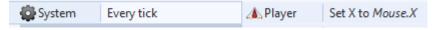
2. Klik tab Event Sheet. Lalu, klik Add Event untuk menambahkan kode program.



3. Pilih System dan pilih Every Tick.



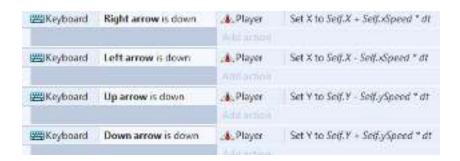
4. Klik Add Action yang ada di bagian kanan dan pilih Player. Lalu, pilih Set X untuk Mouse.



5. Klik tombol Run untuk menguji permainan.



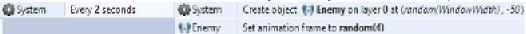
6. Klik kanan pada Player-Set X to Mouse.X dan pilih toggle disable untuk menonaktifkan kode tersebut. Lalu, tambahkan kode untuk kontrol keyboard.



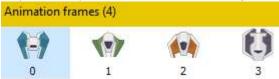
# 5.2.4 Langkah 4-Memunculkan Object Enemy dan Laser

Enemy akan muncul dengan kecepatan vertikal dari atas ke bawah dan koordinatnya random serta objeknya akan berganti-ganti. Langkah untuk menambahkan Event dan Action sama seperti langka sebelumnya.

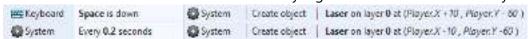
1. Tambahkan kode untuk memunculkan objek Enemy secara acak setiap 2 detik.



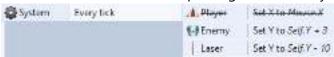
2. Objek Enemy ditampilkan sesuai dengan frame yang telah dibuat sebelumnya



3. Tambahkan kode untuk memunculkan Laser yang akan ditembakkan oleh Player.



4. Tambahkan kode untuk kecepatan gerakan Enemy.



## 5.2.5 Langkah 5-Menambahkan Particle

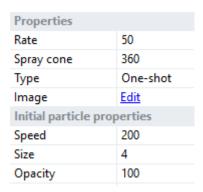
Particle akan membuat atmosfir *gameplay* perang pesawat luar angkasa terasa lebih nyata dengan menampilkan kehancuran dari sebuah objek menjadi ukuran partikel kecil yang menyebar ke segala arah.

1. Klik kanan pada Layout dan pilih Insert New Object. Tambahkan partikel untuk Player dan Enemy.



2. Klik kedua objek partikel tersebut dan ubah property untuk Spray Cone, Type dan Size sesuai gambar di bawah.

#### GameCapstone



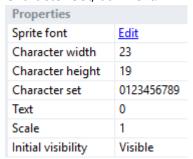
# 5.2.6 Langkah 6-Menambahkan HUD dan Global Variable

HUD atau *Heads-Up Display* adalah tampilan antarmuka pada permainan yang menunjukkan status seperti nyawa dan skor. HUD yang akan ditambahkan adalah Life dan Score.

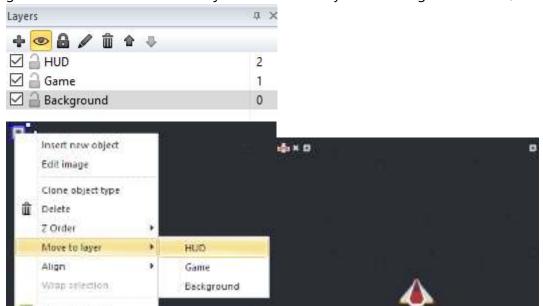
1. Klik kanan pada Layout dan pilih Insert New Object. Tambahkan Indicator, Numeral, Life, dan Score. Perhatikan, khusus untuk Life dan Score ikon yang ditambahkan adalah Sprite Font karena mereka adalah status yang nilainya berubah-ubah tergantung kondisi yang ada.



2. Pilih Life dan Score, lalu ubah properti untuk Character Height, Character Width, Character Set, dan Text.



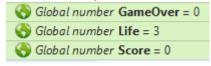
3. Tambahkan Layer untuk HUD dan Game. Lalu, susun tampilan HUD pada Layout seperti gambar di bawah. Pindahkan objek HUD tadi ke Layer HUD dengan klik kanan, mov



4. Pada tab Event Sheet, klik kanan lalu pilih Add global variable.



5. Tambahkan variabel Life untuk nyawa, Score untuk skor, dan GameOver untuk menandakan permainan berakhir.

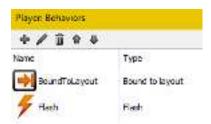


# 5.2.7 Langkah 7-Memberikan Efek Particle Saat Collision

Efek Particle ini akan muncul ketika terjadi *collision* atau tabrakan antara satu objek dengan objek lainnya. Collision yang dimunculkan adalah antar Laser dengan Enemy dan Player dengan Enemy.

1. Klik kanan pada Player dan pilih Behavior. Lalu, tambahkan Flash sebagai efek respawn ketika nanti objek Player hancur.

#### GameCapstone



2. Klik kanan pada Enemy, lalu pilih Instace Variable. Tambahkan HP sebagai darah.



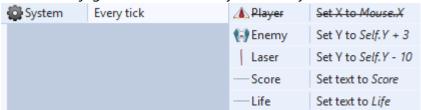
3. Tambahkan kode untuk *collision* antara Laser dengan Enemy dan Player dengan Enemy. Klik kanan pada tanda panah hijau, lalu pilih Add sub event blank untuk menambahkan event tambahan HP = 0 dan GameOver = 0 di bawah event *collision*.



4. Tambahkan event baru di posisi paling atas untuk mengatur variable dengan nilai yang sudah ditentukan.



5. Tambahkan juga action di bawah System-Every Tick.



6. Tambahkan kode untuk variabel terkakhir yang menandakan ketika nyawa habis, maka permainan akan di *restart*.



# 5.2.8 Langkah 8-Memberikan Efek Parallax pada Background

Efek Parallax adalah efek yang membuat Background berjalan terus menerus (*looping*) sehingga terlihat seakan-akan latar belakang permainan berjalan tanpa henti.

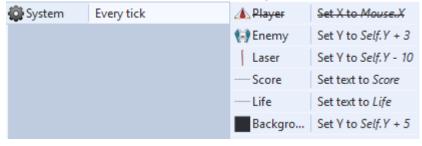
1. Salin Background yang telah ada, lalu tempel di bagian atas.



- 2. Cari koordinat posisi bawah pada Background pertama dan koordinat posisi atas pada Background kedua sehingga ketika Background pertama berjalan dan habis akan dilanjutkan Background kedua, lalu diulang terus.
- 3. Tambahkan kode untuk menetapkan koordinat tadi.



4. Tambahkan kode untuk kecepatan Background



5. Jalankan untuk menguji permainan.

## Referensi

- Bagus, M. (2015, February 26). *Kumpulan Framework untuk Membuat Game dengan HTML5, Bagian 1*. Diambil kembali dari CodePolitan: https://www.codepolitan.com/kumpulan-framework-untuk-membuat-game-dengan-html5-bagian-1
- Fakry, M. (2012, October 29). *Mengenal Tipikal Game Browser Indonesia Lebih Dalam*. Diambil kembali dari KotakGame: http://www.kotakgame.com/feature/detail\_feature/263/34/Mengenal-Tipikal-Game-Browser-Indonesia-Lebih-Dalam
- GameDev. (2017, Feb 01). *How to make a Space Shooter Construct 2 Course*. Diambil kembali dari

  YouTube: https://www.youtube.com/playlist?list=PL0DZV6c9xy7dkoRe2XupAcbnXZrHclTfv
- Group, H. W. (2011, April 05). *HTML5 differences from HTML4*. Diambil kembali dari W3C: https://www.w3.org/TR/2011/WD-html5-diff-20110405/
- Gullen, A. (2011, June 26). *Panduan Pemula Menggunakan Construct 2*. Diambil kembali dari Scirra Construct 2: https://www.scirra.com/tutorials/37/panduan-pemula-menggunakan-construct-2/id/page-1#h2a2
- Scirra. (2007). *What is Construct 2?* Diambil kembali dari Scirra: https://www.scirra.com/construct2
- Sidhion, D. (2012, Feb 08). *Review: Construct 2, a Drag and Drop HTML5 Game Maker*. Diambil kembali dari envato-tuts+: https://code.tutsplus.com/articles/review-construct-2-a-drag-and-drop-html5-game-maker--active-10825

# **Modul 6 - GameCapstone**

# Pengembangan Game Berbasis RPG



# Didukung oleh:



# 6 PENGEMBANGAN GAME BERBASIS RPG

Role-Playing Game (RPG) adalah jenis permainan dengan ciri khas para pemainnya memerankan tokoh-tokoh khayalan dan berkolaborasi untuk merajut sebuah cerita bersama. Mengontrol protagonis atau karakter-karakter dalam sebuah kelompok (biasa disebut *party*) serta menyelesaikan satu atau beberapa misi adalah ciri-ciri penting dari RPG. Para pemain memilih aksi tokoh-tokoh mereka berdasarkan karakteristik tokoh tersebut, dan keberhasilan aksi mereka tergantung dari sistem peraturan permainan yang telah ditentukan. Asal tetap mengikuti peraturan yang ditetapkan, para pemain bisa berimprovisasi membentuk arah dan hasil akhir permainan ini (Harrigan & Wardrip-Fruin, 2010).

Dalam sebuah permainan RPG, jarang ada yang kalah atau menang. Ini membuat permainan RPG berbeda dengan jenis permainan lainnya. Seperti sebuah novel atau film, permainan RPG mempunyai daya tarik karena permainan ini mengajak para pemain untuk menggunakan imajinasi mereka membangun cerita. RPG biasa lebih mengarah ke kolaborasi sosial daripada kompetisi. Pada umumnya dalam RPG, para pemain tergabung dalam satu kelompok. RPG sendiri telah berevolusi dari permainan konsol-jendela berbasis teks sederhana, menjadi permainan dalam bentuk tampilan *third person* sampai menjadi permainan dalam bentuk visual 3D (Hitchens & Drachen, 2007).

Berbicara tentang teknik pengembangannya, salah satu perangkat lunak populer yang dapat dimanfaatkan untuk pembuatan *game* berbasis RPG adalah **RPG Maker MV**.

#### **Tujuan Modul**

- 1. Memahami karakteristik game RPG.
- 2. Mengenali fitur-fitur utama dalam RPG Maker MV.
- 3. Memahami keunggulan dan kekurangan RPG Maker MV dalam pengembangan permainan
- 4. Mampu membuat aplikasi permainan dengan RPG Maker MV.

# 6.1 Ringkasan Pembelajaran

Role-Playing Game (RPG) adalah salah satu jenis permainan yang paling digemari di seluruh dunia. Dalam *game* ini, pemain akan berperan menjadi seorang tokoh di dunia fantasi, berpetualang merekrut tokoh-tokoh lain untuk bertarung bersama, menaikkan level dan kekuatan tempur, serta masuk ke dalam cerita yang sangat menyentuh (Wikugame, 2013).

Sebenarnya, apa yang membuat sebuah *game* RPG begitu menarik? Ada banyak sekali alasannya seperti plot yang dihadirkan dengan sangat baik, kesempatan untuk mengembangkan karakter, musik yang membangun atmosfer, visualisasi, dan tema dunia fantasi. RPG memberikan kesempatan bagi pemain untuk merasakan pengalaman secara langsung untuk terlibat di dalam dunia yang selama ini hanya muncul dalam cerita dan khayalan. Singkatnya, pemain bisa berperan menjadi seorang pahlawan (Santoso, 2012).

Berikut daftar beberapa *game* dengan tema RPG yang telah sukses di pasaran seperti Pokemon Deluge, Zelda, Final Fantasy, The Elder Scrolls, Mass Effect, Valkyrie Profile, Dragon Age Origins, Chrono Series, Seiken Densetsu, Dragon Quest, Fallout, dan Genso Suikoden.



Gambar 6. 1 Game Deluge RPG Pokemon (DelugeRPG, 2007)

#### 6.1.1 Pengenalan RPG Maker MV

RPG Maker adalah perangkat lunak yang dikembangkan untuk membuat sebuah *game* bergenre RPG dan bisa dimanfaatkan oleh pengguna sebagai metode belajar pengembangan *game* RPG. Tanpa kemampuan *coding* atau gambar, pengguna bisa menciptakan *game* RPG sendiri mengikuti alur logika yang sudah disediakan oleh *engine*.

RPG Maker yang dapat digunakan pada Windows adalah RPG Maker 95, RPG Maker 2000, RPG Maker 2003, RPG Maker XP, RPG Maker VX, RPG Maker VX Ace, dan yang paling baru adalah RPG Maker MV.



Gambar 6. 2 RPG Maker MV

Berikut spesifikasi sistem yang dibutuhkan untuk dapat menjalankan RPG Maker MV:

	Windows (standalone/Steam)	Mac (Steam)	Linux (Steam)
OS	WindowsR 7/8/8.1/10 (32bit/64bit)	Mac OS X 10.10 or better	Ubuntu 14.04 or Steam OS 2.0 (64 bit)
CPU	Intel Core2 Duo or better		
RAM	2GB or more		
HDD	Installation needs over 2GB		
Graphics	DirectX 9/OpenGL 4.1 capable GPU	OpenGL 4.1 capable GPU	
Display	1280x768 or better		

Gambar 6. 3 Spesifikasi RPG Make MV

#### 6.1.2 Fitur Game Development dalam RPG Maker MV

Beberapa fitur yang ada pada RPG Maker MV antara lain adalah:

- Dapat digunakan untuk mengembangkan *game* RPG di berbagai *platform* seperti Windows/EXE, MacOS/APP, Android/APK, iOS/IPA, HTML 5 for Web Browser.
- Dapat memilih tampilan mode pertarungan dalam *front-view battle* untuk kamera depan dan *side-view batlle* untuk melihat pertarungan dengan musuh dengan kamera tampak samping.
- Dukungan terhadap kontrol layar sentuh dan mouse.
- Peningkatan batas maksimal basis data.
- Pembagian peta menjadi tiga lapisan.
- Dukungan terhadap resolusi tinggi. RPG Maker VX Ace yang merupakan versi sebelumnya hanya mendukung resolusi 640 x 480.

#### **6.2 Video Tutorial URL**

Contoh game RPG yang akan dikembangkan dengan RPG Maker MV adalah RPG Universe dengan gameplay yang berisi desain map RPG yang menunjukkan lokasi seperti rumah, kastil,

toko dalam bentuk peta, dialog interaksi antara *hero* (karakter yang dipakai oleh pengguna) dan NPC (*Non-Playable Character*), perpindahan tempat dari peta ke suatu tempat, dan *mode battle* dengan musuh.



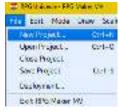
Gambar 6. 4 Tampilan Gameplay RPG Universe

Tautan video tutorial: <a href="https://youtu.be/pqHWloOdD5q">https://youtu.be/pqHWloOdD5q</a>.

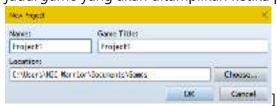
# 6.2.1 Langkah 1-Membuat New Project

Ketika pertama kali menjalankan RPG Maker MV, pengguna akan mendapatkan layar kosong karena belum ada *project* yang pernah digunakan. Untuk dapat mendesain *game* RPG, pengguna harus membuat *project* baru.

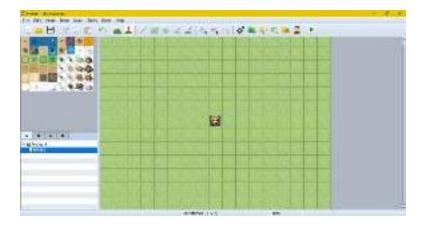
1. Buka RPG Maker MV. Klik tab Menu File, pilih New Project.



2. Isikan kolom **Name** dengan nama *project* yang akan dibuat dan **Game Title** dengan judul *game* yang akan ditampilkan ketika permainan dimulai.



3. Ketika *project* telah selesai memuat semua aset gambar, RPG Maker MV akan menampilkan tampilan World Editor seperti di bawah ini.



#### 6.2.2 Langkah 2-Mendesain Map

Map adalah peta penunjuk lokasi dalam *game* RPG. Peta ini berisi gambaran lokasi seperti rumah, kastil, toko, dan lainnya yang akan membawa *hero* menuju ke tempat tersebut untuk memulai petualangan.

1. **Tile Editor** yang ada di panel bagian kiri berisi *tile* untuk mendesain *map*. Buka masing-masing *tab* A, B, C, dan D untuk melihat jenis *tile* lainnya.



2. Pastikan tombol **Map** pada menu *toolbar* aktif (bukan tombol *Event*) untuk dapat mendesain map.



3. Pilih *tile* yang diinginkan, lalu desain *map* sesuai rancangan peta yang diinginkan.



- 4. Letakkan hero di ujung kiri atas sebagai titik awal keberangkatan.
- 5. Tekan tombol **Playtest** untuk menguji permainan. Lihat, ada sedikit kesalahan ketika *hero* melewati pepohonan hutan karena pengaturan *collision* untuk *tile* pohon memungkinkan *hero* bisa melewatinya.



6. Tekan tombol **Database** pada *toolbar* untuk mengubah pengaturan RPG. Lalu, pilih menu **Tilesets** untuk mengubah *collision* pohon. Jika tandanya masih terdapat O artinya tidak ada *collision*, klik tanda tersebut untuk menggantinya menjadi tanda X. Klik tombol **Apply**, lalu OK.



7. Uji permainan lagi dan lihat sekarang hero tidak bisa melewati pohon.

#### 6.2.3 Langkah 3-Menambahkan NPC (Non-Playable Character)

NPC merupakan salah satu elemen penting dalam *game* RPG karena mereka memberikan informasi mengenai petualangan yang sedang dijalankan seperti NPC yang menunjukkan petunjuk ketika terdapat *puzzle*, NPC yang dapat menjual item kepada *hero*, dan lainnya.

1. Aktifkan tombol **Event** pada *toolbar*.



2. Klik salah satu lokasi koordinat pada **World Editor** untuk meletakkan NPC sehingga layar menampilkan kotak **Event Editor**. Klik **Image**, pilih gambar karakter dari daftar yang ditampilkan. Lalu, klik OK.



3. Di kotak **Contents**, klik 2x pada titik segi empat sehingga kotak Event Commands muncul. Lalu, pilih tombol **Show Text**.



4. Klik 2x pada kotak Face, lalu pilih jenis tampilan muka dari karakter yang diinginkan. Kemudian, tambahkan kalimat pada kotak Text sebagai dialog percakapan ketika nanti karakter NPC di klik saat permainan dimulai.



5. Tambahkan juga Show Choices untuk menampilkan pilihan dan jawaban dari masing-masing pilihan.



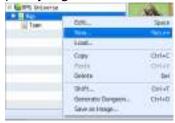
6. Klik tombol Playtest untuk menguji permainan.



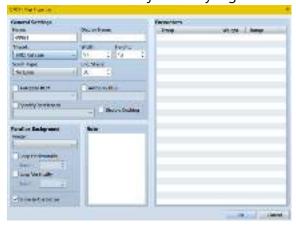
# 6.2.4 Langkah 4-Menambahkan Transfer World

*Transfer World* atau berpindah tempat dalam *game* RPG digunakan untuk memindahkan karakter *hero* ke suatu lokasi seperti toko, rumah, kastil, dan lainnya dengan tampilan dunia yang baru.

1. Di panel kiri bawah Tree, klik kanan pada **Map**, lalu pilih **New** untuk menambahkan peta yang baru.



2. Isikan **Name** untuk memberikan nama *map* yang baru, **Display Name** untuk menampilkan nama peta ketika *hero* masuk ke *map* tersebut. Klik tombol drop-down *Tileset* untuk memilih jenis *tile* yang berbeda.



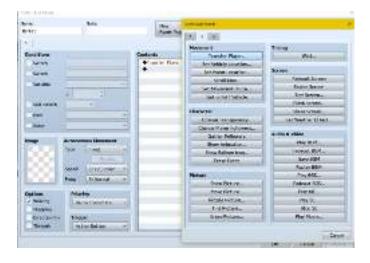
3. Desain *map* yang baru dengan *tile* yang berbeda. Contoh, *map* baru yang akan dibuat adalah pemukiman rumah dengan nama Town.



4. Kembali ke *map* utama, aktifkan tombol **Event** dan klik lokasi koordinat pada bagian peta rumah yang telah dibuat sebagai titik masuk *hero* ke dalam *map* pemukiman rumah tadi.



5. Isikan nama titik tersebut dan klik 2x pada bagian *Contents*, lalu pilih tombol *Transfer Player* untuk memindahkan *hero*.



6. Di *Direct Designation*, klik tombol ... Lalu, pilih map Town dan pilih lokasi koordinat di mana *hero* akan muncul ketika terjadi berpindah tempat.



7. Kembali ke *map* Town, lalu ulangi langkah 4-6 untuk memberikan titik lokasi koordinat untuk berpindah tempat dari pemukiman rumah kembali ke *map* utama.

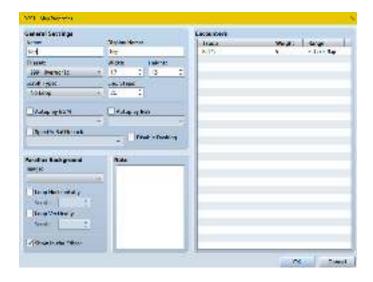
#### 6.2.5 Langkah 5-Menambahkan Mode Battle

Mode Battle atau pertarungan dalam RPG Maker MV sudah diatur secara otomatis, tetapi pengaturan mulai dari darah, musuh, serangan, dan lainnya dapat diubah. Salah satu contoh mode battle yang sering digunakan adalah ketika hero berjalan-jalan di map utama atau map lainnya dan secara otomatis akan berganti layar menuju pertarungan dengan musuh.

1. Di panel kiri bawah *Tree*, klik kanan pada nama **Map**, lalu pilih *Edit*.



2. Di kotak *Enc. Steps* atur berapa langkah yang diperlukan oleh *hero* ketik berjalan-jalan di *map* utama untuk memasuki *mode battle*. Di bagian kanan *Encounter*, klik kanan, lalu pilih *Edit* untuk menambahkan *Troops* (kumpulan musuh).



3. Klik tombol **Database** dan pilih menu *System* untuk mengubah pengaturan pertarungan. Pengguna dapat menghapus beberapa karakter yang mengikuti *hero* utama dan menambahkan opsi *mode battle* dalam tampilan samping dengan klik centang *Use side-view battle*.



4. Playtest untuk menguji permainan.



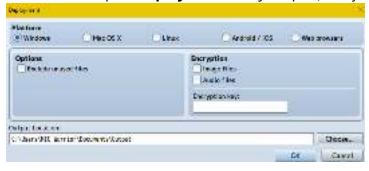
5. Tambahkan beberapa elemen game RPG lainnya. Misal, seperti mode Boss.



# 6.2.6 Langkah 6-Mengekspor Project

Ekspor *project* sehingga *game* RPG dapat dimainkan pada *platform* Windows, Mac, Mobile, dan lainnya.

1. Klik menu **File**, pilih **Deployment**. Pilih jenis *platform* yang diinginkan. Lalu, klik OK.



#### Referensi

- DelugeRPG. (2007). *About DelugeRPG*. Diambil kembali dari DelugeRPG: https://www.delugerpg.com/about
- Fahmi, M. (2015, August 04). *RPG Maker MV Diumumkan! Buat RPG Sendiri dan Rilis untuk Smartphone!* Diambil kembali dari Tech in Asia Indonesia: https://id.techinasia.com/rpg-maker-mv-dukung-resolusi-tinggi-dan-smartphone
- Harrigan, P., & Wardrip-Fruin, N. (2010). *Second Person: Role-playing and Story in Games and Playable Media*. Cambridge: MIT Press.
- Hitchens, M., & Drachen, A. (2007). The Many Faces of Role-Playing Games. *International Journal of Role-Playing*, 3-21.
- Santoso, P. (2012, February 10). *10 Monster yang Wajib Hadir di Game RPG*. Diambil kembali dari JagatPlay: http://jagatplay.com/2012/02/news/10-monster-yang-wajib-hadir-digame-rpg/
- Wikugame. (2013). *10 Game RPG Terbaik Sepanjang Masa*. Diambil kembali dari Wikugame: http://www.wikugame.com/2013/08/10-game-rpg-terbaik-sepanjang-masa.html

# **Modul 7 - GameCapstone**

# Pengembangan Game Berbasis 3D



# Didukung oleh:



# 7 PENGEMBANGAN GAME BERBASIS 3D

Game berbasis 3D merupakan hiburan komputer yang interaktif yang dipresentasikan dalam bentuk tiga dimensi, yang memiliki tinggi, lebar, dan kedalaman. Perbedaan yang dapat terlihat dari game 3D dibandingkan dengan 2D ialah adanya kedalaman yang memberikan kemampuan lebih dalam *game* terutama mengenai eksplorasi dunia virtual dengan representasi yang lebih realistis. (Rouse, 2017)

Bahkan pada saat ini, *game* 3D menjadi jenis *game* yang lebih banyak disukai karena lebih luas dibandingkan *game* 2D. Pengembangan *game* 3D dapat dilakukan dengan beberapa *engine* yang tersedia, salah satunya yaitu dengan Unity.

#### **Tujuan Modul**

- 1. Mengenali fitur-fitur utama dalam Unity.
- 2. Memahami komponen dan struktur pengembangan aplikasi dan mampu menggunakan aset bawaan Unity.
- 3. Mampu menggunakan Scripting dan Physics untuk pemrograman pada Unity
- 4. Mampu menggunakan UI dan Animation untuk artist pada Unity
- 5. Mampu membuat permainan sederhana dengan Unity.

# 7.1 Ringkasan Pembelajaran

Game berbasis 3D merupakan perkembangan dari game berbasis 2D, perbedaan yang paling mencolok ialah dengan adanya penambahan satu dimensi yang berupa kedalaman. Penambahan dimensi ini menyebabkan adanya kemungkinan untuk menambahkan gerakan dan peletakan objek di dalam game. Tetapi, setiap objek dan karakter yang dibutuhkan dalam pembuatan game berbasis 3D membutuhkan art yang lebih kompleks dibandingkan dengan game berbasis 2D.

Saat ini, sudah terdapat banyak *game engine* atau perangkat lunak yang dapat digunakan untuk melakukan pengembangan *game* 3D, salah satunya ialah *game engine* Unity.

# 7.1.1 Pengenalan Unity

Unity bukanlah hanya berupa *game engine* yang ditujukan untuk membuat *game*, tetapi juga merupakan suatu *game engine* yang terbentuk dari komunitas. Dalam website <u>unity3d.com</u>, terdapat forum, tutorial, pasar *asset*, *show case*, dan hal lainnya yang berhubungan dengan *game engine* Unity. (Games, 2017)



Gambar 7. 1 Tampilan Unity (Technologies, 2017)

Unity tidak hanya ditujukan untuk pengembang *game* tertentu, tetapi juga untuk pengembang *game* berskala kecil atau *startup* yang hanya terdiri dari 1 hingga 5 orang. Hal ini dikarenakan Unity memberikan berbagai jenis tawaran yang dapat digunakan, yaitu (Technologies, 2017):

# 1. Personal - Harga: Free

Tawaran ini ditujukan bagi pengembang tingkat dasar, pelajar dan para penggembar yang ingin mengeksplorasi dan baru mencoba menggunakan Unity. Meskipun tawaran ini bersifat gratis, keseluruhan fitur utama dalam Unity sudah dapat digunakan.

#### 2. Plus - Harga: \$35 per bulan

Ditujukan untuk pengembang tingkat menengah yang sudah serius untuk mewujudkan ide-ide mereka dan berencana untuk mempublikasikannya. Tawaran ini juga memberikan beberapa kelebihan lainnya, antara lain:

- a. Splash Screen yang dapat dikustomisasi.
- b. Laporan performa game.
- c. Pengaturan fitur multiplayer hingga 50 pemain pada satu waktu.
- d. Pro Skin Editor UI.
- e. Diskon hingga 20% untuk pembelian asset dalam asset store Unity.
- 3. *Pro -* Harga: \$125 per bulan

Tawaran ini ditujukan bagi pengembang profesional yang membutuhkan tingkat fleksibilitas lebih tinggi dan membutuhkan kustomisasi lebih lanjut. Beberapa kelebihan lainnya dengan menggunakan tawaran ini, antara lain:

- a. Layanan tingkat *pro*, termasuk *Cloud Build*, penambahan pemain dalam fitur *multiplayer*, dan *insight* yang lebih dalam.
- b. Tidak terdapat restriksi mengenai pendapatan yang bisa dihasilkan.
- c. Diskon hingga 40% untuk pembelian asset dalam asset store Unity.
- 4. Enterprise Harga: fleksibel / sesuai kebutuhan

Tawaran ini ditujukan bagi pengembang profesional yang sudah cukup besar, yaitu terdiri dari 21 anggota atau lebih. Tawaran ini memberikan kelebihan bagi peengembang dengan tambahan bantuan secara langsung dengan Unity.

Unity merupakan *game engine* yang ditujukan untuk pengembangan *game* 3D yang sesuai dengan anggaran yang tersedia. Bahkan tawaran *personal* yang digunakan dapat merilis game secara *commercial*. (Widder, 2017). Berikut spesifikasi sistem yang dibutuhkan untuk menjalankan Unity untuk pengembangan (Technologies, 2017):

- OS: Windows 7 SP1+, 8, 10; Mac OS X 10.8+.
   Windows XP & Vista tidak didukung; dan versi server Windows & OS X tidak diuji.
- 2. GPU: Graphics card with DX9 (shader model 3.0) atau DX11 with feature level 9.3 capabilities.
- 3. Spesifikasi lainnya bergantung pada tingkat kompleksitas dari project yang dikerjakan.

Kebutuhan untuk pengembangan platform tambahan:

- 1. iOS: Mac dengan minimal OS X 10.9.4 version dan Xcode 7.0 atau lebih tinggi.
- 2. Android: Android SDK dan Java Development Kit (JDK).
- 3. Windows Store: Windows 8.1 (64-bit) dan Visual Studio dan *platform* SDK seperti berikut:
  - a. Windows 8.1/Windows Phone 8.1: Visual Studio 2013 atau versi sebelumnya dan Windows 8.1 SDK;
  - b. UWP: Visual Studio 2015 atau lebih tinggi dan Windows 10 SDK.
  - c. IL2CPP *scripting backend* juga membutuhkan fitur C++ *compiler* terinstal dalam Visual Studio.
- 4. WebGL: Mac OS X 10.8+ atau Windows 7 SP1+ (64-bit editor only).

# 7.1.2 Fitur Game Development pada Unity

Versi terbaru dalam Unity ialah Unity 5.6.2 yang dirilis pada tanggal 21 Juni 2017, kemudian versi Unity 2017 yang dirilis pada tanggal 10 Juli 2017. Beberapa fitur unik pengembangan *game* pada Unity, antara lain (Technologies, 2017):

- Rich & extensible editor.
  - Unity versi tebaru memberikan tambahan berupa *creative hub* untuk *artist, designer, developer* dan tim member lainnya. Termasuk *design tools* dalam 2D dan 3D *scene, instant play mode* untuk *rapid editing* dan iterasi, dan sistem animasi yang *powerful*.
- *Graphics Rendering.*Memberikan *atmosphere* yang dapat memikat pemain. Dengan adanya tampilan cahaya terang, neon, cahaya redup, dan bayangan untuk membuat *game* menjadi lebih dinamis.
- Engine Performance.
   Unity masih terus dikembangkan agar memiliki performa yang lebih tinggi lagi.
  - Industry-leading multiplatform.

    Game yang dikembangkan cukup satu saja dan nanti sistem Unity akan membantu untuk men-deploy ke banyak platform. Secara keseluruhan terdapat 25 platforms lebih dari platform yang dapat di-build menggunakan Unity, termasuk mobile, desktop, console, TV, VR, AR dan Web.



Gambar 7. 2 Plaform yang dapat di-build menggunakan unity (Technologies, 2017)

#### Unity Asset Store

Unity memiliki *asset store* berisi banyak *asset* yang dapat membantu dalam pengembangan *game*.



Gambar 7. 3 Unity Asset Store (Technologies, 2017)

#### - Multiplayer

Unity memberikan kemudahan dalam pengembangan *multiplayer*, baik pembuatan *real time*, atau pun *networked game* 

#### - Team Collaboration

Unity *Team* memungkinkan pengembang untuk bekerja bersama, sehingga dapat meningkatkan efisiensi dengan menggunakan *version control* atau *cloud storage*.

#### Unity Connect

Unity *Connect* merupakan pasar dalam mencari jaringan profesional dan *talent* yang didedikasikan untuk Unity *creators*.

#### - Live Ops Analytics

Unity *Analytics* memberikan pengembang akses yang cepat dan mudah mengenai informasi penting yang membantu untuk mengembangkan ekonomi *in-game* dan pengalaman pemain.

#### - Performance Optimization

Pelaporan performa Unity diberikan secara *real time*. Pelaporan ini dapat membantu untuk menemukan *error* atau *bugs* sehingga dapat diatasi dengan lebih cepat.

#### Monetization

Unity menawarkan solusi *built-in* untuk meningkatkan kesukesan pengembang *game*. Beberapa solusi yang ditawarkan Unity adalah *Ads (advertisements)* dan *iAP (in-app Purchase)* 

#### Advertising

Unity menawarkan setiap pengembang *game* untuk mengiklankan *game* mereka melalui Unity *Ads*.

#### - Viral user growth

Dengan menggunakan *Everyplay*, pemain dapat membagikan video mengenai *best moments* mereka dan memamerkannya ke teman mereka.

# 7.2 Video Tutorial URL

*Game* yang akan dikembangkan berupa *game racing* sederhana dengan menggunakan Unity. Unity yang digunakan dalam tutorial adalah Unity versi 5.6.x.



Gambar 7. 4 Tampilan Game Racing

Tautan Video Tutorial: <a href="https://youtu.be/u8Z7BUU3JGw">https://youtu.be/u8Z7BUU3JGw</a>.

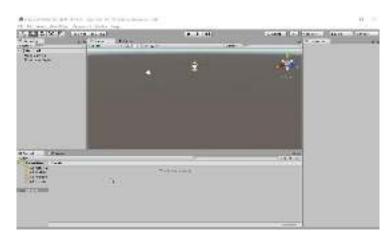
#### 7.2.1 Langkah 1-Membuat New Project

Ikuti langkah berikut untuk membuat projek baru Unity,



Gambar 7. 5 Pembuatan New Project

- 1. Buka Unity kemudian klik tombol 'NEW'.
- 2. Ubah 'Project Name' menjadi 'Project Car'.
- 3. Pastikan 3D terpilih.
- 4. Klik pada tombol 'Create Project'.
- 5. Berikut tampilan project baru pada Unity.

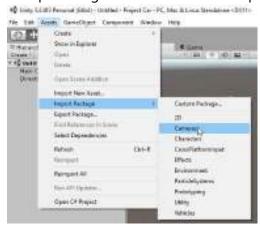


Gambar 7. 6 Tampilan Awal Unity

# 7.2.2 Langkah 2-Meng-Import Packages

Unity sudah menyediakan banyak *standard asset* yand dapat digunakan untuk membantu pengembangan *game*, ikuti langkah berikut untuk meng-*import* beberapa *package* yang dibutuhkan dalam tutorial ini:

1. Pilih pada bagian menu 'Assets -> Import Package'.

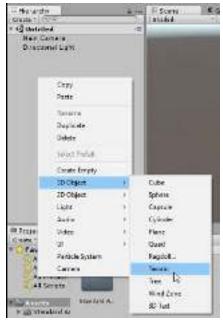


2. Kemudian pilih 'Cameras', 'Environment', dan 'Vehicles'.

# 7.2.3 Langkah 3-Membuat *Track*

Track yang digunakan dibuat menggunakan object terrain yang disediakan dari Unity.

- 1. Klik kanan pada bagian 'Hierarchy'.
- 2. Pilih '3D Object -> Terrain'.



3. Kemudian pada object Terrain bagian 'Inspector' klik bagian 'Setting'.



Lalu ubah bagian:

'Resolution -> Terrain Width' menjadi '200'.

'Resolution -> Terrain Heigth' menjadi '200'.

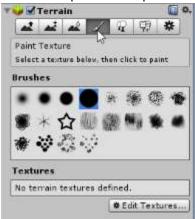


Setelah itu ubah bagian 'Transform -> Position' menjadi (-100, 0, -100).

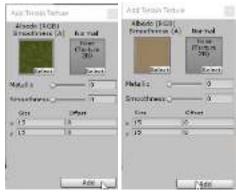
Pengubahan posisi ini ditujukan untuk memposisikan Terrain dibagian tengah (0, 0, 0).



4. Kemudian pindahkan pada bagian 'Paint Texture'.



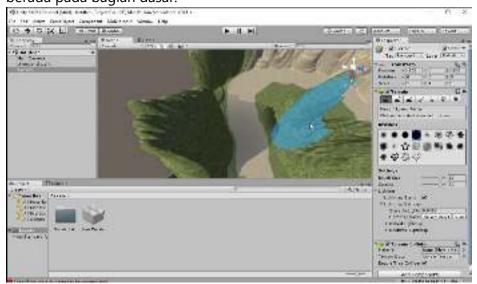
Tambahkan 'Textures' dengan mengklik bagian 'Edit Textures... -> Add Texture'. Lalu tambahkan 'Terrain Texture' dua kali, dengan *texture* pertama untuk keseluruhan *terrain* dan *texture* kedua untuk mewarnai jalan.



Lalu warnai *terrain* sehingga membentuk sebuah lintasan / *track* balap, misalnya seperti berikut:



Lalu tambahkan pembatas di setiap ujung *terrain* agar mobil tidak terjatuh dari lintasan, menggunakan 'Raise / Lower Terrain'. Klik kiri untuk menaikan *terrain* dan klik kiri + *shift* untuk menurunkan *terrain*, secara *default* ketika *terrain* dibuat, *terrain* berada pada bagian dasar.



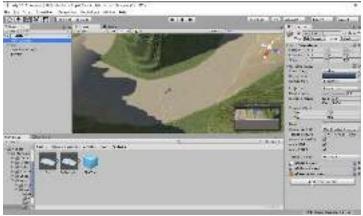
### 7.2.4 Langkah 4-Menambahkan Karakter

*Game* yang dirancang merupakan *game racing* maka karakter yang akan ditampilkan adalah kendaraan mobil.

1. Tambahkan mobil menggunakan *prefab* yang sudah disediakan dari *standard asset*. Prefab *car* dapat ditemukan di 'Standard Assets -> Vehicles -> Car -> Prefabs'. *Drag & drop* prefab *car* ke dalam lintasan, jangan lupa untuk mengubah posisi dan rotasi sesuai dengan kebutuhan.



2. Kemudian pada bagian 'Main Camera' klik 'Add Component'.



Pilih 'Script -> UnityStandardAssets.Cameras -> Auto Cam'



3. Kemudian tambahkan 'object' kosong untuk menjadi target 'main camera' Tambahkan objek berupa 'Create Empty' pada objek 'Car'.



Lalu ubah nama objek tersebut menjadi 'ObjectCamera' dan ubah 'Transform -> position' menjadi '(0, 4, -8).



- 4. Kemudian pada objek 'Main Camera' bagian 'Auto Cam', ubah beberapa *properties* yang ada sebagai berikut:
  - 'Target' menjadi 'ObjectCamera'; drag & drop 'ObjectCamera' ke dalam kolom 'Target'.
  - 'Auto Target Player' menjadi 'false'.
  - 'Update Type' menjadi 'Late Update'.
  - 'Turn Speed' menjadi '1.5'.
  - 'Spin Turn Limit' menjadi '360'.



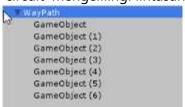
# 7.2.5 Langkah 5-Menambahkan Karakter Al

Karakter Al berupa mobil sama seperti karakter pemain.

1. Tambahkan mobil menggunakan *prefab* yang sudah disediakan dari *standard asset*. Prefab *car* dapat ditemukan di 'Standard Assets -> Vehicles -> Car -> Prefabs'. *Drag & drop* prefab *CarWaypointBased* ke dalam lintasan, jangan lupa untuk mengubah posisi dan rotasi sesuai dengan kebutuhan.



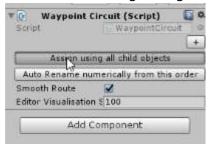
2. Kemudian tambahkan objek kosong 'Create Empty' seperti pada sebelumnya, dan tambahkan beberapa objek kosong 'Create Empty' pada objek tersebut. Berikan Objek Utama nama 'WayPath'. Ubah posisi setiap 'GameObject' sehingga membuat sebuah 'circuit' mengelilingi lintasan.



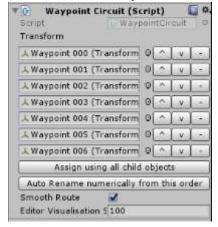
3. Kemudian pada bagian *Inspector*, tambahkan *script* 'Waypoint Circuit'. Pilih 'Add component -> Script -> UnityStandardAssets.Utility -> Waypoint Circuit'



Kemudian klik 'Assign using all child objects'.



Ubah posisi sesuai urutan perjalanan dalam *circuit*, jika sudah, klik pada bagian 'Auto Rename numerically from this order'.



4. Lalu pada objek 'CarWaypointBased' bagian 'Inspector' temukan *script* 'Waypoint Progress Tracker' dan *drag & drop* objek 'WayPath' ke dalam bagian 'Circuit'



5. Jika sudah, maka tampilan scene akan terlihat sebagai berikut.



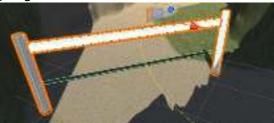
#### 7.2.6 Langkah 6-Menambahkan Garis Finish dan Check Point

Game racing membutuhkan garis finish untuk menentukan apakah lintasan sudah selesai dijalani atau belum, dan dibutuhkan check point untuk memastikan pemain berjalan melalui lintasan yang sudah disediakan.

1. Buatlah sebuah tempat untuk merepresentasikan garis. Tambahkan objek kosong 'Create Empty' dan tambahkan 4 'Cube' ke dalamnya.



Dengan 2 'Cube' sebagai pilar, 1 'Cube' sebagai bagian atas, dan 1 'Cube' sebagai pengecekan.



a. Bagian pilar



Ubah 'Scale' pada 2 'Cube' menjadi (1, 12, 1). Posisikan kedua 'Cube' sehingga saling sejajar dan terdapat jarak.

 Bagian atas
 Posisikan dan ubah ukuran 'Cube' sehingga menjadi bagian atas dari kedua pilar yang sudah dibuat sebelumnya.

c. Bagian pengecekan

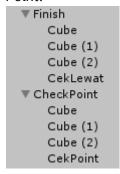


Posisikan 'Cube' ini di bagian bawah / bagian mobil lewat. Pastikan ukuran 'Cube' memanjang dari satu pilar ke pilar lainnya, dan buat 'Cube' memiliki tinggi paling tidak setinggi mobil. Lalu pastikan 'Box Collider -> Is Trigger' bernilai 'true'.



Kemudian hilangkan tampilan 'Cube' dengan melakukan klik kanan pada 'Mesh Renderer' lalu pilih 'Remove Component'

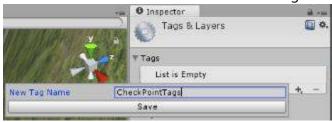
2. Ulangi langkah satu dan letakkan di bagian tengah lintasan, sebagai tempat *Check Point*.



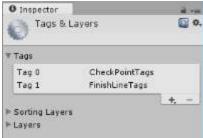
3. Tambahkan 'Tag' dengan mengklik bagian 'Tag' di Inspector dan pilih 'Add Tag...'.



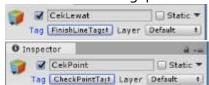
Lalu klik tombol '+' untuk menambahkan 'Tag' baru.



Tambahkan 2 'Tag' dan berikan nama yang sesuai seperti gambar di bawah ini lalu klik 'Save', "CheckPointTags' dan 'FinishLineTags'.



Kemudian ubah 'Tag' pada setiap 'Cube' pengecek yang disiapkan



#### 7.2.7 Langkah 7-Menambahkan Script untuk Melakukan Pengecekan

Objek yang sudah dipersiapkan tidak akan berfungsi jika belum dituliskan *script* untuk memperogramnya.

1. Pada bagian objek 'Car' klik 'Add Component -> New Script'. Lalu beri nama 'HitungCekPoint' kemudian klik 'Create and Add'.



2. Kemudian pada bagian folder 'Asset' klik dua kali pada 'HitungCekPoint.cs'.



3. Masukkan kode berikut agar pemrograman UI dapat dilakukan.

```
using UnityEngine.UI;
```

4. Masukkan kode berikut untuk mempersiapkan variabel dasar yang akan digunakan dalam *game*.

```
bool lewatCheckPoint = false;
public int jumlahLap = 2;
int lapSaatIni = 0;
public Text lapTeks;
public Text finishTeks;
```

5. Masukkan kode berikut untuk memberikan teks pada 'lapTeks'.

```
void Start ()
{
    lapTeks.text = lapSaatIni + " / " + jumlahLap;
}
```

6. Tambahkan kode berikut untuk melakukan pengecekan ketika mobil karakter mengenai garis *finish* atau *check point*.

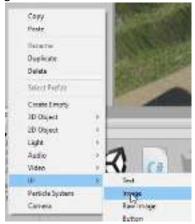
```
void OnTriggerEnter (Collider garisCheck)
{
    if (garisCheck.tag == "CheckPointTags") {
        if (lewatCheckPoint == false) {
            lewatCheckPoint = true;
        }
    }
}
```

```
if (garisCheck.tag == "FinishLineTags") {
    if (lewatCheckPoint == true) {
        if (lapSaatIni < jumlahLap) {
            lapSaatIni += 1;
            lapTeks.text = lapSaatIni + " / " + jumlahLap;
        } else
            finishTeks.text = "Selamat Anda sudah menyelesaikan seluruh lap!";
        lewatCheckPoint = false;
    }
}</pre>
```

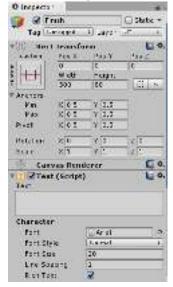
#### 7.2.8 Langkah 8-Menambahkan UI Text untuk Menampilkan Tulisan pada Layar

*Script* yang dibuat masih belum dapat digunakan dikarenakan belum memiliki UI *text* untuk menerima dan menampilkan teks dari *script*.

1. Klik kanan pada bagian 'Hierarchy' kemudian pilih 'UI -> Text'. Tambahkan 2 *text* pada *game*.

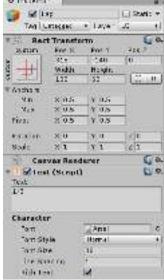


2. Berikan pengubahan 'properties' pada salah satu UI text sebagai berikut.



- 'Width dan Height' menjadi '300, 80'.
- 'Text' menjadi ' ' (kosongkan kolom Text).
- 'Character -> Font Size' menjadi '20'.
- 'Color' menjadi berwarna putih.

3. Berikan pengubahan 'properties' pada UI *text* lainnya sebagai berikut.

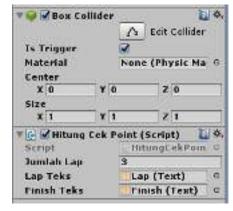


- 'Position' menjadi '315, -140, 0'.
- 'Width dan Height' menjadi '100, 60'.
- 'Text' menjadi '1/3' (untuk mempermudah tampilan ketika melakukan editing).
- 'Character -> Font Size' menjadi '36'.
- 'Paragraph -> Alignment' menjadi 'middle, middle'.
- 'Color' menjadi berwarna putih.

#### 7.2.9 Langkah 9-Menyempurnakan Script yang Sudah Dipersiapkan

Script 'Hitung Cek Point' pada objek 'Car' masih belum dapat digunakan.

- 1. Tambahkan komponen 'Box Collider' pada objek 'Car'. 'Add Component -> Physics -> Box Collider'. Pastikan 'Is Trigger' bernilai 'true'.
- 2. Kemudian *drag & drop* kedua *text* sebelumnya ke dalam *script* sesuai dengan nama yang sudah diberikan.



#### 7.2.10 Langkah 10-Uji Coba permainan

Uji coba permainan untuk memastikan seluruh script sudah dituliskan dengan benar

1. Ujicoba dapat dilakukan dengan menekan tombol 'play' pada bagian atas.



# 7.2.11 Langkah 11-Mem-build game

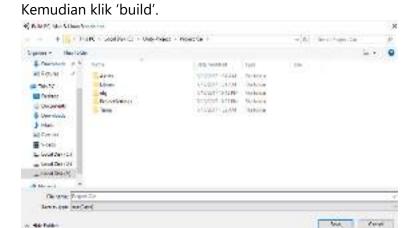
Game sudah berjalan dengan baik, waktunya menggunakan fitur build game sehingga game dapat dimainkan tanpa menggunakan Unity.

1. Klik 'File -> Build Settings'.



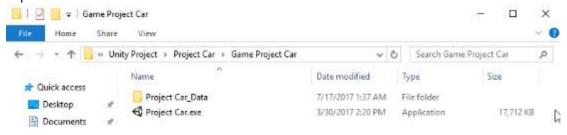
2. Kemudian tambahkan *scene* ke dalam *'Scene In Build'*. Klik 'Add Open Scenes'. Jika diminta menyimpan *scene*, simpanlah *scene* dengan nama yang diinginkan.





Lalu berikan nama sesuai yang diinginkan, dan klik 'save'.

3. Ketika proses *build* selesai dijalankan, dalam *folder* tersebut akan bertambah berkas seperti berikut.



# 7.2.12 Langkah 12-Selamat bermain dan berbagi

Selamat!! *Game* sudah dapat dimainkan dan dapat dibagikan. *Game* dapat dibagikan dengan membagikan kedua berkas yang merupakan hasil *build*.



#### Referensi

- Games, R. (2017). *Unity Game Engine Review*. Diambil kembali dari Game Sparks: https://www.gamesparks.com/blog/unity-game-engine-review/
- Rouse, M. (2017). *3D Gaming*. Diambil kembali dari What is Tech Target: http://whatis.techtarget.com/definition/3D-gaming
- Technologies, U. (2017). Unity 3D. Diambil kembali dari Unity 3D: unity3d.com
- Widder, B. (2017, Mei 8). *How to make a video game (experience not required)*. Diambil kembali dari Digital Trends: https://www.digitaltrends.com/gaming/how-to-make-a-video-game/2/