BAB III

LANDASAN TEORI

Pada bab ini, akan dibahas teori, bahasa pemrograman, *platform*, serta *software* yang digunakan untuk membuat *game Creatures Adventure*.

3.1. *Game*

Game adalah sistem dimana pemain berpartisipasi dalam konflik buatan yang didefinisikan dengan aturan-aturan yang menghasilkan hasil yang dapat diukur (Salen & Zimmerman, 2003). *Game* memiliki beberapa jenis dasar, antara lain (Genre Definitions, n.d.):

3.3.3. Action

Gambar 3.1. Grand Theft Auto V

Jenis *game* ini membutuhkan kemampuan mekanik antara salah satu atau lebih dari hal berikut:

- Akurasi
- Gerakan
- Keputusan yang cepat
- Refleks
- Pemilihan waktu

Jenis *game* ini hanya digunakan pada *game* yang jenisnya tidak cocok untuk sebagai jenis *racing*, *role-playing*, dan *sports*.

3.3.4. *Adventure*

Gambar 3.2. Dishonored 2

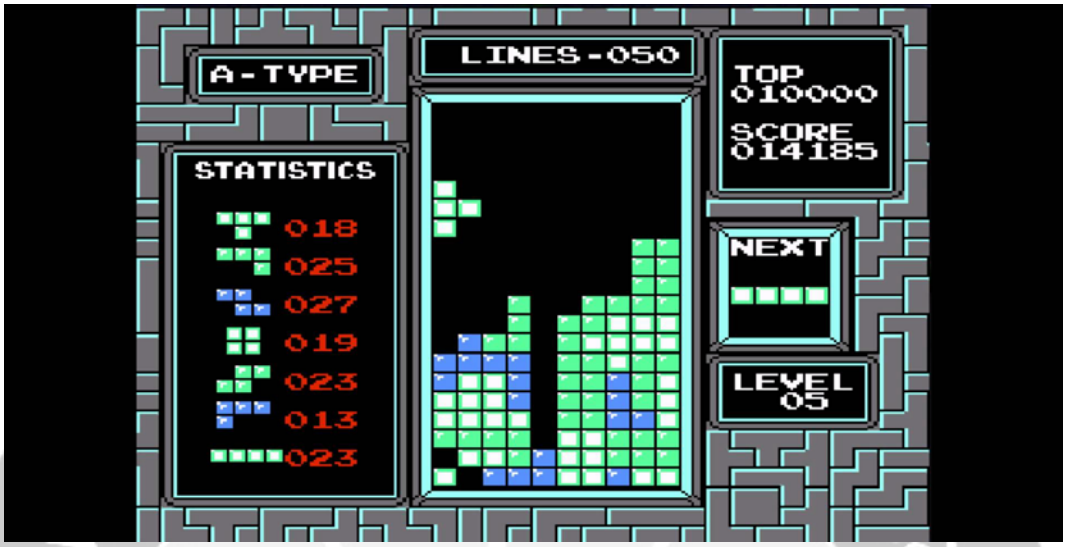
Jenis *game* ini menekankan untuk mengalami suatu cerita melalui dialog dan pemecahan teka-teki. Mekanik dari *game* menekankan pada pemilihan dibandingkan aksi. Pemecahan teka-teki biasanya berkisar tentang penggabungan atau pemanipulasian barang-barang untuk melanjutkan cerita.

3.3.5. Educational

Gambar 3.3. Balloony Word

Jenis *game* ini mencoba mengajari pemain melalui *game* dan biasanya ditujukan untuk anak-anak. Jenis *game* ini juga menawarkan rasa menarik, sebagai cara yang tidak langsung untuk mempraktekkan subyek atau mata pelajaran yang kadang dianggap tidak menarik seperti pengejaan, matematika, sejarah, dan lain-lain.

3.3.6. Puzzle



Gambar 3.4. Tetris

Jenis *game* ini hanya berfokus untuk memecahkan teka-teki yang biasanya tidak memiliki banyak cerita. *Puzzle* muncul dalam berbagai macam, antara lain strategik, taktikal, *logical*, *trivia*, dan lain-lain.

3.3.7. Racing / Driving

Gambar 3.5. DiRT Rally

Jenis *game* yang mengijinkan pemain untuk melakukan balapan atau mengendarai kendaraan dengan tidak tergesa-gesa. Balapan dapat dilakukan pada kendaraan, tunggangan, berjalan, atau pada grafik yang abstrak.

3.3.8. Role-Playing (RPG)

Gambar 3.6. Final Fantasy VII

Jenis *game* ini termasuk pada jenis *game* yang berfokus pada perkembangan karakter. Beberapa aspek tambahan yang biasa d*item*ukan pada RPG antara lain mengumpulkan kekayaan, cerita, dan pertarungan taktikal. Perkembangan karakter tidak harus terjadi seperti pada cerita-cerita tradisional, tetapi dapat juga terjadi ketika karakter mempelajari kemampuan baru atau meningkatkan kemampuan sebelumnya. Perkembangan karakter juga termasuk mengumpulkan peralatan atau keperluan yang dapat meningkatkan kekuatan secara bertahap.

3.3.9. Simulation

Gambar 3.7. The Sims 3

Jenis *game* ini memiliki beberapa tipe, tetapi hal yang dimiliki secara umum adalah jenis *game* ini dibuat secara lebih nyata dengan kehidupan nyata dibandingkan jenis *game* lainnya. Beberapa tipenya antara lain:

- *Business/Trade Simulations*
- *Construction Simulations*
- *Life Simulations*
- *Management Simulations*
- *Sports Simulations*
- *Vehicle Simulators and Vehicular Combat Simulators*
- *War Simulations*

3.3.10. Sports

Gambar 3.8. NBA 2K17

Jenis *game* ini adalah *game* dimana pengguna mengendalikan pemain atau manajer dari olahraga yang nyata atau fiksional.

3.3.11. Strategy

Gambar 3.9. Civilization V

Jenis *game* ini berkisar antara penggunaan strategik dan/atau taktikal dari sumber daya yang biasanya ada pada pertarungan atau skenario yang berhubungan dengan pengelolaan.

Game strategi merupakan *game* dimana pemain menjadi pengambil keputusan bagaimana unit yang diatur akan bergerak. Tujuan dari jenis *game* ini adalah untuk membawa pemain ke dalam dunia *game* yang memberikan mereka sejumlah keputusan untuk dibuat, bersama dengan alasan yang kuat kenapa mereka membuat keputusan tersebut dan menghasilkan konsekuensi (Ayangbekun & Akinde, 2014).

3.2. *User Interface*

User interface merupakan salah satu hal yang paling penting pada sistem atau produk yang berbasis komputer. Jika *user interface* dirancang dengan kurang baik, kemampuan *user* untuk memanfaatkan kekuatan komputasi dari suatu aplikasi mungkin akan sangat berkurang. Kenyataannya, *interface* yang lemah

dapat mengakibatkan aplikasi yang telah dirancang dengan baik dan telah diimplementasikan menjadi gagal (Sridevi, 2014).

Ada delapan aturan emas dalam perancangan *user interface*, antara lain (Shneiderman & Plaisant, 2005):

1. **Berusaha untuk konsisten**

   Rangkaian aksi yang konsisten dibutuhkan pada aksi yang sama; istilah yang identikal harus digunakan pada prompts, menu, dan layar bantuan; dan perintah yang sama harus selalu dipergunakan.

2. **Memungkinkan pengguna yang sering menggunakan aplikasi untuk menggunakan *shortcuts***

   Dengan bertambahkan frekuensi penggunaan aplikasi, pengguna ingin untuk mengurangi jumlah interaksi dan meningkankan kecepatan interaksi.

3. **Menawarkan umpan balik yang informatif**

   Untuk setiap aksi yang dilakukan pengguna, harus ada umpan balik dari sistem. Untuk aksi yang kecil dan sering terjadi, umpan balik bisa sederhana, sedangkan untuk aksi yang besar dan jarang terjadi, umpan balik harus lebih jelas.

4. **Merancang dialog untuk menghasilkan penutupan**

   Rangkaian aksi harus diatur ke dalam grup dengan pembuka, bagian tengah, dan akhir. Umpan balik yang informatif pada penyelesaian aksi memberi kepuasan kepada pengguna dan mengindikasikan bahwa pengguna sudah dapat mempersiapkan aksi selanjutnya.

5. **Menawarkan penyelesaian kesalahan sederhana**

   Semampu mungkin, rancanglah sistem sehingga pengguna tidak dapat membuat kesalahan yang serius. Jika kesalahan terjadi, sistem harus dapat mendeteksinya dan menawarkan penyelesaian yang sederhana.

6. **Mengijinkan pembalikan aksi yang mudah**

   Fitur ini meringankan kegelisahan karena pengguna tahu bahwa kesalahan dapat diperbaiki ke keadaan sebelumnya, yang memberanikan pengguna untuk mencoba pilihan yang tidak biasa dicoba.

7. Mendukung fokus kontrol internal

Pengguna yang berpengalaman sangat menginginkan agar mereka memiliki bagian dalam sistem dan sistem merespon aksi mereka. Rancang sistem yang membuat pengguna sebagai inisiator dari aksi daripada sebagai perespon.

8. Mengurangi beban memori jangka pendek

Adanya batas dari pemrosesan informasi manusia pada memori jangka pendek membutuhkan tampilan yang sederhana, beberapa halaman dapat digabungkan, dan gerakan *window* dapat dikurangi.

3.3. *Tools* Pengembangan

3.3.1. Unity3D

Unity3D merupakan *game engine / authoring tools* yang dapat mempermudah *game designer* dalam membuat *game*, bahasa pemrograman yang bisa digunakan pada unity3d adalah *javascript, c# script* dan *boo*, pada pembuatan *game* dengan unity3d, setiap *level* didefinisikan sebagai sebuah *scene*, dimana *scene* tersebut merupakan area yang bisa diakses oleh pemain ketika *user* memainkan *game* (Fauzi, Cahyana, & Tresnawati, 2013).

Unity3D juga secara otomatis melakukan sinkronisasi dengan MonoDevelop sebagai *tool coding*-nya. Unity3D mendukung lingkungan *multi-platform* seperti Windows, Mac, Xbox360, PS3, Wii, Android, IOS, Explorer, Safari, dan Chrome, serta dapat dijalankan pada OS Windows dan Mac (Bae & Kim, 2014).

3.3.2. C#

C# adalah sebuah bahasa pemrograman modern tingkat tinggi untuk mengembangkan suatu aplikasi dengan menggunakan Visual Studio dan .Net Framework. C# merupakan bahasa pemrograman yang sederhana, kuat, aman dan menggunakan konsep *object oriented programming* (Waruwu, 2015).

3.3.3. PHP

PHP adalah bahasa pemrograman bersifat *server-side* yang didesain secara khusus untuk situs. PHP merupakan produk *open source*, yang artinya *source code* dapat diakses dan digunakan, dimodifikasi, serta didistribusikan secara

bebas. Pada Agustus 2004, PHP telah dipasang pada lebih dari 17 juta *domain* di seluruh dunia, dan angka ini terus berkembang secara cepat (Welling & Thomson, 2005).

Lima karakteristik penting PHP antara lain adalah simplisitas, efisiensi, keamanan, fleksibilitas, dan familiritas (PHP, 2016). PHP biasanya berada dalam HTML dan dituliskan dalam tag <?php…?> seperti contoh potongan kode berikut akan menghasilkan tulisan “Hello, World!”:

```
<html>
<head>
<body>
   <?php echo "Hello, World!";?>
</body>
</head>
</html>
```

3.3.4. MySQL

MySQL merupakan *relational database management system* (RDBMS) yang cepat. *Server* MySQL mengontrol akses dari data untuk menjamin beberapa pengguna dapat menggunakannya secara bersamaan, untuk membuat akses yang cepat, serta menjamin hanya pengguna yang berwenang yang dapat mendapatkan aksesnya. MySQL menggunakan Structured Query Language (SQL) yang merupakan bahasa *query* standar di seluruh dunia (Welling & Thomson, 2005).