Akbar, F. F., Asriyanik, A. and Az-Zahra, F. F. (2023) "Implementasi Objek & Lintasan Looping dengan Metode Object Pooling pada Game Caffeine Rush", Sebatik, 27(1).
Submitted: 2023-04-06 Accepted: 2023-05-28 Published: 2023-06-06

Sebatik Vol. 27 No. 1 Juni 2023
ISSN: 1410-3737(p) 2621-069X(e)
Open access article licensed under CC-BY
DOI:10.46984/sebatik.v27i1.2236

IMPLEMENTASI OBJEK & LINTASAN *LOOPING* DENGAN METODE *OBJECT POOLING* PADA *GAME CAFFEINE RUSH*

Fadhil Faizal Akbar[1)], Asriyanik[2)], dan **Fathia Frazna Az-Zahra[3)]**

[1,2,3]Teknik Informatika, Universitas Muhammadiyah Sukabumi
[1,2,3] Jl. R. Syamsudin, S.H. No. 50, Cikole, Kec. Cikole, Sukabumi, Jawa Barat, 43113
E-mail: fadhildesanta@gmail.com[1)], asriyanik263@ummi.ac.id[2)], fathiafrazna@ummi.ac.id [3)]

ABSTRAK

Perkembangan era teknologi saat ini telah terjadi banyak perubahan yang sangat pesat, salah satunya adalah perkembangan *game* di seluruh dunia. Sudah banyak sekali *game* yang dibuat dan juga sudah banyak variasi dari jenis *game* itu sendiri, salah satu contohnya adalah *game* bergenre *cafein rush*. *Game Caffeine Rush* sendiri menerapkan bentuk menggunakan metode *Game Development Life Cycle* sebagai bentuk pemodelannya, lalu implementasi *instantiate & destroy* karena dapat memunculkan lalu menghancurkan *game object*. Akan tetapi, implementasi tersebut belum sepenuhnya berjalan dengan baik karena dapat menghambat kinerja CPU serta menumpuknya penyimpanan sampah (*cache*). Namun, ada metode yang lebih baik dalam mengoptimalkan pelaksanaan pembuatan dan penghapusan objek secara masif yaitu dengan metode *object pooling*, dimana perbedaannya pada metode ini menggunakan cara mengaktifkan dan menonaktifkan objek. Sehingga dengan cara ini akan mengoptimalkan permainan agar performanya tidak turun dan penggunaan memori tidak terlalu memakan banyak tempat.

Kata Kunci: *Game, Game Object, Object Pooling, Instantiate & Destroy, Game Development Life Cycle, Cache*

1. PENDAHULUAN

Perkembangan teknologi di era digital sekarang telah berkembang sangat pesat salah satunya dalam dunia *game*. Dimulai dari *game arcade* yang menggambarkan versi 8-bit sampai dengan 3D, kini game dapat dimainkan di PC, *console* maupun smartphone. Seiring dengan perkembangan zaman game memiliki variasi *genre* seperti *MMORPG, Racing, Tower Defense, Battle Royal First Person Shooting, Endless Runner*, dan masih banyak lagi (Pratama, 2022).

Game merupakan kegiatan yang melibatkan keputusan pemain dalam berupaya mencapai tujuan dengan dibatasi oleh konteks tertentu (Momoda, 2020). Mengenai sebuah *game*, Indonesia menempati posisi kedua dibawah negara Filipina dalam urutan pemakaian untuk bermain *game*, terutama menggunakan android sebagai platform bermain *game*. Dilansir dari laman Kompas Tekno pada *"Digital 2022 April Global Statshot Report"*, perangkat *smartphone* seperti lebih sering dimainkan dengan total persentase mencapai 67 persen yang menempati urutan pertama, kemudian *PC/laptop* sebanyak 37 persen di urutan kedua, dan konsol *PlayStation* dan *Xbox* sebanyak 25,3 persen di urutan ketiga (Septiyani, 2022). Setelah melakukan kajian penelitian berupa kuesioner, pengguna lebih sering memainkan *game* berbasis *android* karena mudah dibawa dan juga bisa dimainkan dimana saja, salah satunya *game* berbasis *endless runner* menjadi pilihan utama dalam memasarkan produk-produk tertentu (Krisdiawan, 2020).

Endless runner menjadi salah satu genre *game* yang populer saat ini karena bersifat berkelanjutan tanpa berujung, artinya permainan tidak akan berhenti jika pemain dapat bertahan hidup melewati rintangan-rintangan yang ada demi meraih skor tertinggi, dan aturan-aturan pada game endless runner tidaklah rumit sehingga mudah dimainkan. Contohnya seperti *Subway Surfers, Temple Run*, serta *Geometry Dash* (Apriyandi, 2019).

Setelah mengetahui studi kasus diatas, alasan penulis menggunakan *game Caffeine Rush* sebagai *sample* penelitian ini karena *game* ini merupakan *prototype game endless runner* bertemakan biji kopi yang menggelinding secara terus menerus dan meraih skor tertinggi sebaik mungkin. *Game Caffeine Rush* ini mengimplementasikan *instantiate & destroy* pada objek seperti lintasan, rintangan, serta *coin* secara terus menerus. Akan tetapi belum sepenuhnya optimal karena bersifat membuat ulang, sehingga mengakibatkan penumpukan sampah (*cache*) pada memori dan menghambat kinerja CPU ketika menjalankan sebuah *game* (Pekonen, 2019). Untuk menjawab solusi permasalahan ini, *Object Pooling* merupakan solusi metode dalam mengoptimalkan *instantiate & destroy* yang sebagaimana ia akan berada di dalam kolam (*pool*) tersebut dengan cara penggunaan kembali pada objek yang hancur atau hilang (Tan, 2021). Selain itu, *object pooling* juga merupakan cara untuk mengoptimalkan proyek dan menurunkan beban yang ditempatkan pada CPU ketika melakukan pemanggilan cepat pada saat membuat dan menghancurkan objek pada *game*, sehingga menghilangkan beban memori (*cache*) yang tidak diperlukan (Koulaxidis, 2022).

© 2023, The Author(s). Authors retain all their rights to the published works, such as (but not limited to) the following rights; Copyright and other proprietary rights relating to the article, such as patent rights, The right to use the substance of the article in own future works, including lectures and books, The right to reproduce the article for own purposes, The right to self-archive the article

Lalu terdapat jurnal penelitian terdahulu yang berjudul "Analisis Penerapan Metode *Object Pooling* pada *Game 2d Endless Runner*" menjadi tolak ukur dalam penelitian ini. Karena penelitian terdahulu ini menjelaskan mengenai *game* berbasis *endless runner* dalam memunculkan serta menghilangkan objek pada *game* secara terus-menerus, serta menerapkan metode *Object Pooling* sebagai bentuk optimalisasi pada game tersebut.

Merujuk pada penelitian sebelumnya oleh Aditya Cipta Rahardian, Iwan Setiawan, dan George Pri Hartawan mengenai *game endless runner*, penelitian ini diharapkan dapat menawarkan solusi yang efektif dalam mengurangi beban memori (*cache*) serta meningkatkan kinerja CPU pada ketika game dijalankan.

2. RUANG LINGKUP

Setelah menyimpulkan beberapa permasalahan, maka terdapat ruang lingkup antara mengenai *game* berbasis *endless runner* dengan metode implementasi *instantiate & destroy* yang sebelumnya belum optimal karena dapat meningkatkan beban (*cache*) dalam memunculkan serta menghilangkan objek pada game, sehingga dapat menghambat kinerja CPU ketika game dijalankan.

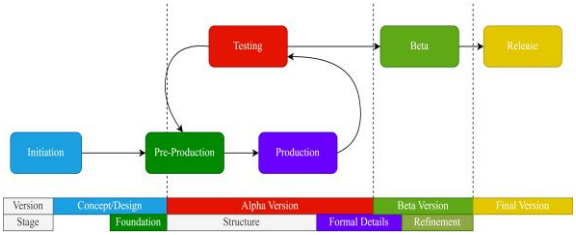
3. BAHAN DAN METODE

Bahan dan metode yang digunakan pada penelitian ini dijelaskan sebagai berikut:

3.1 Metode *Game Development Life Cycle (GDLC)*

Game Development Life Cycle adalah metode pemodelan mengenai penanganan rancangan *game*, dimulai dari tahap awal hingga akhir sebuah rancangan *game* itu sendiri (Sugiarto, 2022). Tujuannya agar *game* yang dibuat sesuai dengan apa yang dirancang baik untuk pembuat maupun target penggunaanya. Sebagaimana yang dijelaskan pada Gambar 1, *Game Development Life Cycle* memiliki beberapa tahap, yaitu:

1. *Initiation*
   Menentukan ide serta konsep *game* yang akan dibuat, misalnya seperti tema *game*, *target audience*, serta jenis *game* yang akan dirancang.
2. *Pre-Production*
   Tahap perancangan desain prototipe *game* seperti mock-up tampilan UI/UX, serta *Game Mechanic*.
3. *Production*
   Tahap membuat prototipe *game*, hasil yang sudah dibuat pada tahap *Pre-Production* akan diterapkan dan dikembangkan menjadi sebuah aplikasi prototipe *game*.
4. *Alpha Test*
   Pada tahap ini, dilakukan sebuah pengujian game yang dilakukan oleh pembuat dengan tujuan apakah ada sebuah kerusakan (*error*) atau kekurangan yang ada pada *game*. Jika merasa sudah sesuai apa yang diharapkan, maka akan dilanjutkan pada tahap selanjutnya.
5. *Beta Test*
   Tahap ini serupa dengan *Testing*, perbedaannya adalah yang menguji prototipe *game* akan dilakukan oleh *Beta Tester*. Pemain harus mencari sebuah kesalahan serta kekurangan yang ada pada *game* tersebut yang nantinya menjadi bahan kritik serta saran untuk pengembangan *game* kedepannya.



Gambar 1. Tahap pemodelan *Game Development Life Cycle*

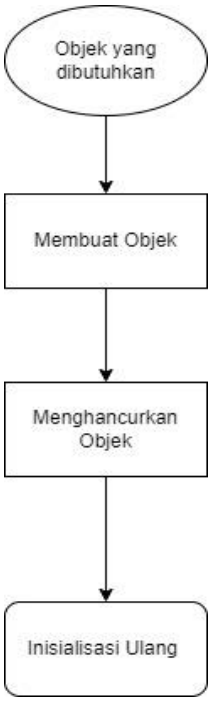
3.2 Metode *Instantiate & Destroy*

Merupakan implementasi bersifat "membuat dan menghapus" *game objects* yang terjadi sepanjang waktu di hampir setiap *game*. *Instantiate & destroy* adalah bentuk dasar ketika ingin membuat sebuah *game* berbasis *endless runner* (Krouvi, 2019). Akan tetapi metode ini belum sepenuhnya optimal karena bersifat membuat ulang daripada menggunakan ulang pada objek *game* yang ada.

Menurut Gambar 2 dijelaskan mengenai cara kerja *Instantiate & Destroy* sebagai berikut:

1. Menyiapkan objek yang dibutuhkan untuk melakukan inisialisasi ulang
2. Setelah itu, objek akan dibuat di dalam game tersebut.
3. Objek akan hancur setelah digunakan atau dilalui.
4. Objek yang hilang akan dilakukan inisialisasi ulang kembali untuk menciptakan sebuah objek yang baru.

Langkah kerja ini merupakan cara penginisialisasian ulang hanya dengan membuat dan menghancurkan objek secara berulang. Hanya saja dapat memakan banyak kapasitas jika tidak dioptimalkan.



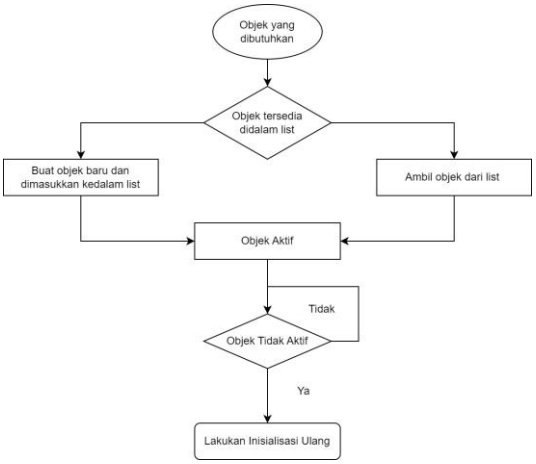
Gambar 2. Cara Kerja implementasi *Instantiate & Destroy*

3.3 Metode *Object Pooling*

Object pooling merupakan metode di mana objek digunakan kembali layaknya mendaur ulang dan menghancurkannya setelah digunakan. Objek diambil dan dikembalikan lagi ke kumpulan setelah digunakan (Rahadian, 2022). Selain itu, metode *object pooling* juga merupakan metode optimalisasi pada *Instantiate & Destroy* karena dapat menghemat pelepasan memori akibat penumpukan sampah (*cache*) serta alokasi memori pada saat pemanggilan/pembuatan objek yang hilang.

Berdasarkan Gambar 3 berikut adalah mengenai cara kerja pada metode *Object Pooling*:

1. Menyiapkan objek yang dibutuhkan untuk melakukan inisialisasi ulang.
2. Membuat sebuah *list* objek yang sudah disiapkan, list tersebut akan bertemu dengan 2 kondisi:
3. Membuat objek baru untuk dimasukan kedalam list. Kondisi ini apabila muncul objek baru yang berbeda dari list yang ada, maka objek baru tersebut akan dimasukan kedalam list.
4. Mengambil objek yang ada pada list. Kondisi ini apabila tidak ada objek yang baru, maka ia akan menggunakan objek yang ada ke tahap berikutnya.
5. Tahapan berikutnya adalah mengaktifkan objek yang ada didalam list.
6. Jika objek tersebut tidak lagi digunakan, maka akan dinonaktifkan untuk dilanjutkan ke tahap selanjutnya.
7. Objek yang sudah tidak aktif akan dilakukan inisialisasi ulang kembali.



Gambar 3. Cara Kerja implementasi Object Pooling

3.4 *C Sharp* (*C#*)

Merupakan bahasa *C++* yang telah dikombinasi oleh fitur-fitur yang ada pada bahasa pemrograman lainnya seperti *Delphi*, *Java*, *Visual Basic*, *Python*, dan sebagainya dengan beberapa penyederhanaan (Perdana, 2021). Pada penelitian ini, penulis menggunakan bahasa *C#* dalam pembuatan *game Caffeine Rush* karena game ini menggunakan *C#* sebagai bahasa pemogramannya pada script *game*-nya.

3.5 *Unity 3D*

Unity 3D adalah sebuah aplikasi membuat *game* atau disebut juga dengan *game engine* yang digunakan untuk membuat game yang bisa dimainkan di berbagai perangkat contohnya *game console*, *Android*, *Windows*, *IOS*, dan lain-lain (Akbar, 2019).

Penulis menggunakan aplikasi *Unity 3D* dalam pembuatan *game*, menyimpan aset, *sprite*, serta *membuat script pemograman game Caffeine Rush* di dalam aplikasi. Selain itu, penulis mengunakan ekstensi tambahan yang disebut *Unity Profiler* dalam menganalisis kinerja CPU, kecepatan FPS (*Frame Per Second*) saat melakukan sebuah *rendering*, serta melihat seberapa kecil atau besar penumpukan sampah (*cache*) yang ada ketika *game* dijalankan.

3.6 *Frame Per Second (FPS)*

Istilah yang mengacu pada jumlah gambaran yang diperbarui per detiknya. Setiap projeksi aset atau tampilan pada *game* memiliki sebuah bingkai (*frame*) yang dimunculkan secara cepat sehingga memunculkan sebuah ilusi ketika dilihat oleh manusia itu sendiri. Semakin besar FPS yang dibuat, maka semakin baik ilusi yang dibuat per detiknya, akan tetapi jika FPS-nya kecil, maka ilusi proyeksi gambar yang dibuatnya akan semakin buruk (Koulaxidis, 2022).

Pengujian FPS ini juga sangat penting dalam menganalisis kinerja CPU *game* dan juga mengetahui apakah ada penumpukan sampah (*cache*) pada memori pada saat *game* dijalan atau sebaliknya.

© 2023, The Author(s). Authors retain all their rights to the published works, such as (but not limited to) the following rights; Copyright and other proprietary rights relating to the article, such as patent rights, The right to use the substance of the article in own future works, including lectures and books, The right to reproduce the article for own purposes, The right to self-archive the article

3.7 *Game Design*

Game Design merupakan suatu proses untuk menciptakan peraturan dan konten yang dapat membuat pemain merasa termotivasi dalam mencapai tujuan dengan mengikuti peraturan yang telah dibuat (Sehang, 2019).

Game design sangat penting dalam proses suatu rancangan *game* ini, misalnya seperti menentukan tema yang akan dibuat, membentuk sebuah mekanisme yang akan diterapkan pada *game* agar membuat sebuah aturan yang menarik untuk dimainkan sesuai dengan tema yang diambil.

3.8 *Game Mechanic*

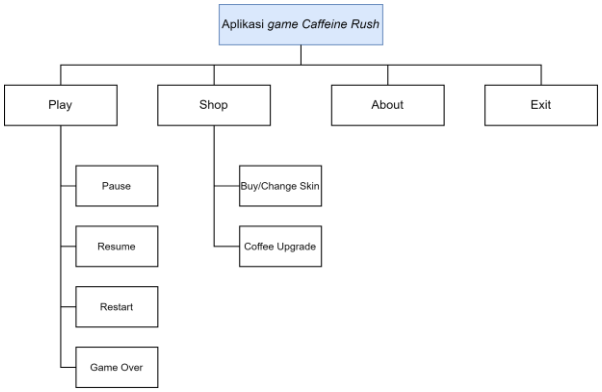
Game Mechanic adalah ragam aturan yang mengatur tindakan dan perilaku pemain serta keadaan (*state*) dalam sebuah *game*. *Game Mechanic* merupakan bentuk jawaban dari pertanyaan-pertanyaan berupa *IF*, *ELSE*, dan *THEN* yang nantinya mengacu bagaimana pemain mengambil tindakan dari permasalahan yang ada pada *game* (Landers, 2022).

4. HASIL DAN PEMBAHASAN

Tahapan pada perancangan game dengan menggunakan teknik permodelan *Game Development Life Cycle*, dimulai dari *initiation* dalam pemilihan tema hingga tahapan *Beta Test*, berikut penjelasan tahapan perancangan game:

4.1 *Initiation*

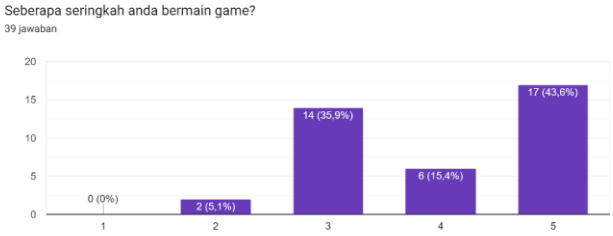
Berdasarkan Gambar 4 ini merupakan struktur aplikasi *game Caffeine Rush* secara menyeluruh. Tema *game Caffeine Rush* sendiri mengambil tema yang berlatar di sebuah kafe atau kedai kopi dengan biji kopi yang menggelinding sebagai karakter utama pada *game*. Untuk target penggunaan pada *game Caffeine Rush* dapat dimainkan untuk semua golongan dari usia 6 tahun hingga usia dewasa 30 tahun ke atas.



Gambar 4. Struktur Menu Rancangan Pada *Game Caffeine Rush*

Untuk melengkapi penelitian ini, penulis membuat sebuah kuisioner berupa pertanyaan mengenai rancangan *game Caffeine Rush*. Kuisioner ini berguna untuk membantu penulis untuk melengkapi penelitian ini seperti seberapa sering orang bermain *game*, berapa lama pemain menghabiskan waktu bermain *game*, serta rasa suka terhadap minuman kopi dan alasannya.

Berdasarkan Gambar 5 berdasarkan hasil pengisian kuisioner dengan skala 1-5, survei mengatakan bahwa 43,6% pemain sering memainkan *game*, sedangkan 15,4% cukup sering memainkannya, 35,9% memiliki tingkat kecenderungan yang netral dalam bermainan *game*, 5,1% kadang-kadang memainkan *game*, dan sisanya tidak ada satupun responden yang mengaku jarang memainkan *game*.



Gambar 5. Diagram Balok Mengenai Seberapa Sering Responden Memainkan Game.

Hasil survei pada **Gambar 6**, hasil menunjukkan bahwa 41% pemain menghabiskan waktu bermain *game* lebih dari 3 jam, 23,1% untuk pemain yang menghabiskan waktu bermain *game* antara 1 jam hingga 3 jam, 10,3% pemain yang hanya menghabiskan waktu bermain *game* sekitar 30 menit, dan 2,6% untuk pemain yang menghabiskan waktu sekitar 10 menit dalam bermain *game*.

Gambar 6 Diagram *Pie* Hasil Survei Berapa Lama Responden Menghabiskan Waktu Bermain *Game*

4.2 *Pre-Production*

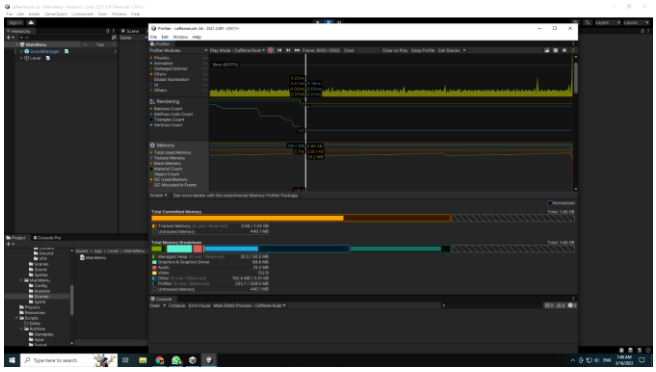
Pre-Production pada *game Caffeine Rush* dibagi menjadi beberapa bagian. Game Design yang terdiri dari *Level Design* dan *User Interface (UI/UX)*, sedangkan pada *Game Mechanic* menjadi 2 bagian terdiri dari Core *Loop* dan *Core Mechanic*.

4.3 *Production*

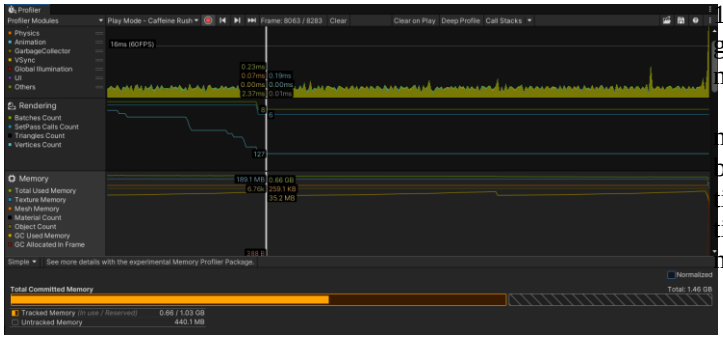
Pada tahap *Production*, penulis mulai membuat aplikasi dari hasil *Pre-Production* yang sudah jadi. Dimulai dari aset, tampilan antarmuka, serta mulai membuat program serta mengimplementasikan *Instantiate & Destroy* dan *Object Pooling* sebagai optimalisasi yang nantinya akan dimasukkan kedalam *script game.* Untuk hasil pada *Production* akan cukup berbeda dengan rancangan yang dibuat pada *Pre-Production.*

4.4 *Alpha Test*

Pada **Gambar 7** ini, penulis melakukan sebuah pengujian *game Caffeine Rush Alpha-Version* yang disebut dengan *black box*. Pengujian ini dilakukan untuk mencari sebuah kerusakan (*error*), pemakaian memori, kecepatan *frame per second* (FPS), serta kinerja memori yang ada pada *game*. Pengujian *black box* ini dilakukan pada aplikasi *Unity 3D* itu sendiri dengan ekstensi *Profiler*.



Gambar 7 Uji Black Box Dengan Unity Profiler.



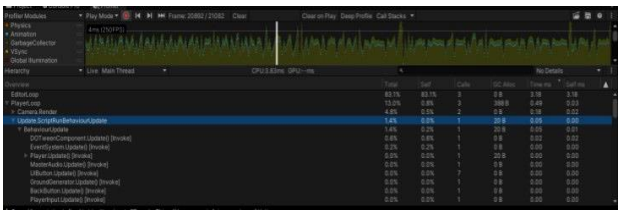
Gambar 8 Hasil analisis kinerja aplikasi pada komputer.

Berikut adalah hasil uji *black box* dalam berbentuk tabel 1

Tabel 1. Hasil Uji Black Box.

No.	Keterangan	Hasil Yang Didapatkan
1.	FPS yang didapatkan	*60 frame per second* (16ms)
2.	Penggunaan RAM	± 600MB
3.	*Penumpukan Memori (Cache)*	± 221KB
4.	Kecepatan dalam meminimalisir *cache*	± 2,37 ms

Berdasarkan Gambar 8 dan Tabel 1, sejauh ini tidak ada tanda-tanda kerusakan atau penumpukan sampah (*cache*) sehingga kinerja CPU berjalan dengan lancar dengan total penggunaan memori RAM ± 600MB, *Cache* ± 221KB. Lalu pada pengujian FPS, hasil menyatakan bahwa *game Caffeine Rush* dapat meraih *60 frame per second* (16ms), sehingga pembuatan *frame* pada aset *game Caffeine Rush* dapat berjalan dengan baik, dan yang terakhir kecepatan meminimalisir *cache* ± 2,37 ms.



Gambar 9 Hasil analisis *object* melakukan *looping* terus menerus.

Lalu pada Gambar 9 dijelaskan bahwa objek pada *game* melakukan pemunculan serta penghancuran *looping* secara *up to date* pada saat *game* dijalankan ± 1.4% dengan pemakaian memori ± 20 Byte yang artinya game tersebut sangat ringan dalam memunculkan serta menghilangkan objek.

Terakhir pada Gambar 10 dijelaskan bahwa objek melakukan pemunculan kembali secara *looping* dan juga berurutan dari 1 hingga 10, yang artinya objek tersebut tidaklah membuat ulang di dalam hirarki *game*, sehingga tidak memakan banyak kapasitas memori serta tidak menghambat kinerja CPU pada saat *game* dijalankan.

Gambar 10. Proses *re-use* pada saat *game* berjalan.

© 2023, The Author(s). Authors retain all their rights to the published works, such as (but not limited to) the following rights; Copyright and other proprietary rights relating to the article, such as patent rights, The right to use the substance of the article in own future works, including lectures and books, The right to reproduce the article for own purposes, The right to self-archive the article

4.5 *Beta Test*

Disini pengujian prototipe *game Caffeine Rush Beta-Version* akan diuji oleh *beta tester* yang akan menguji serta mencari sebuah kesalahan atau kekurangan yang ada pada *game* untuk pengembangan *game* kedepannya. Data tersebut dikumpulkan lalu di masukan ke dalam tabel. Berikut adalah hasil pengujian *game* oleh *beta tester*.

Tabel 2. Hasil pengujian oleh para *beta tester*.

No.	Keterangan	Persentase Positif	Persentase Negatif
1.	Apakah kinerja CPU dapat berjalan dengan lancar?	100% Lancar	0% Tidak Lancar
2.	Apakah ada penumpukan sampah (*Cache*) pada Memori?	66,7% Tidak Ada	33,3% Ada
3.	*Rating Score* (1-5) pada *game Caffeine Rush*	55,6% pada rating 4-5	44,4% pada rating 3
4.	Apakah anda tertarik untuk memainkannya lagi?	94,4% Ya	5,6% Tidak

Berdasarkan hasil pada **Tabel 2** dijelaskan bahwa responden menjawab persentase positif 100% ketika kinerja CPU berjalan dengan lancar. Selanjutnya responden menjawab positif 66,7% bahwa tidak ada penumpukan sampah (cache) pada saat game dijalankan, akan tetapi sebagian menjawab negatif 33,3% bahwa masih ada penumpukan sampah (cache) pada saat *game* dijalankan.

Untuk hasil penilaian, responden menjawab positif 55,6% bahwa *game* ini mendapatkan rating 4-5, akan tetapi sebagian mendapatkan rating 44,4% dengan *rating* 3. Dan yang terakhir responden menjawab positif tertarik untuk memainkannya lagi sebanyak 94,4%, sebagian menjawab negatif bahwa responden tidak tertarik memainkan game ini berkisar 5,6%

5. KESIMPULAN

Metode *Object Pooling* dalam mengoptimalisasi *instantiate & destroy* pada objek lintasan dan rintangan dapat berjalan dengan baik tanpa adanya sebuah penghancuran dengan cara pemanggilan kembali secara terus-menerus pada aset *game Caffeine Rush* dengan total kecepatan *frame* berkisar 60 FPS (16ms). Lalu untuk hasil implementasi *Object Pooling* juga dapat memperlancar proses kinerja CPU pada *android device* tanpa ada proses penundaan (*lag*) ketika menjalankan *game Caffeine Rush* ± 600MB dari penggunaan memori RAM yang diuji. Yang terakhir, implementasi *Object Pooling* pada *game Caffeine Rush* dapat meminimalisir penumpukan sampah (*cache*) pada *android device* pada saat *game* dijalankan ± 221KB dengan kecepatan minimalisir *cache* berkisar 2,37ms.

6. SARAN

Adapula sebuah saran yang penulis simpulkan. Penelitian ini ditujukan dalam mengoptimalisasi objek yang hilang pada saat melakukan pemanggilan ulang dengan metode *object pooling*. Selain itu, *object pooling* dapat meminimalisir penumpukan sampah (*cache*) pada *game* berbasis *endless runner* dengan metode, sehingga dapat dijalankan secara maksimal. Pengujian yang dilakukan pada aplikasi prototipe, sebaiknya dilakukan dengan menggunakan uji *black box* sebelum aplikasi digunakan kepada pengguna awal.

Melalui teknik pengujian *black box* akan menguji fungsionalitas *input* serta *output* sistem pada tingkat tinggi dan pengujian ini memberikan gambaran bagaimana aplikasi berjalan ketika digunakan pertama kali oleh pengguna awal sehingga mendapatkan keputusan yang tepat dalam mengembangkan hingga memperbaiki aplikasi di masa yang akan datang.

7. DAFTAR PUSTAKA

Akbar. (2019, September 9). *Apa itu Unity3D ? | Pengertian Unity 3D | Akbar Project*. Akbar Project Software Developer. https://akbarproject.com/apa-itu-unity-3d/

Apriyandi, D. (2019). Penerapan Endless Runner Game Untuk Memperkenalkan Pariwiasta Kota Pontianak. *Core.Ac.Uk*, *7*(3). https://core.ac.uk/download/pdf/296441652.pdf

Jacky D.Sehang, Virginia Tulenan, & Alwin M.Sambul. (2019). Perancangan Game Simulasi Kewirausahaan. *Jurnal Teknik Informatika*, *14*(1), 1–10.

Jerry Momoda. (2020). *Endless Runner Games: Evolution and Future*. http://jerrymomoda.com/analysis-endless-runners/

Koulaxidis, G. (2022). Improving Mobile Game Performance with Basic Optimization Techniques in Unity. *Mdpi.Com*. https://doi.org/10.3390/modelling3020014

Krisdiawan, R. A. (2020). Pembuatan Game Runaway From Culik Dengan Algoritma Fuzzy Mamdani. *Journal.Uniku.Ac.Id*, *6*(1), 33–39. https://doi.org/https://doi.org/10.25134/buffer.v6i1.2623

Krouvi, S. (2019). Implementation Of A Video Game Collection Games For Educational Use. *South-Eastern Finland University of Applied Sciences*, 1–37.

Landers, R. N. (2022). Game-based, gamified, and gamefully designed assessments for employee selection: Definitions, distinctions, design, and validation. *Wiley Online Library*, *30*(1), 1–13. https://doi.org/10.1111/ijsa.12376

Pekonen, P. (2019). *Fundamentals of HTML5 game*

optimization. https://lutpub.lut.fi/handle/10024/160161

Perdana, M. H. P., Atmaja, P. W., & Wahanani, H. E. (2021). PEMBUATAN GAME TOWER DEFENSE HEROES CONQUEST MENGGUNAKAN UNITY. *Prosiding Seminar Nasional Informatika Bela Negara*, *2*, 78–87. https://doi.org/10.33005/santika.v2i0.112

Pratama, D. (2022). *Subway Surfers Jadi Mobile Game Paling Banyak Diunduh di Seluruh Dunia*. https://www.liputan6.com/tekno/read/4970232/subway-surfers-jadi-mobile-game-paling-banyak-diunduh-di-seluruh-dunia

Rahadian, A., … I. S.-P. J. I., & 2022, undefined. (2022). ANALISIS PENERAPAN METODE OBJECT POOLING PADA GAME 2D ENDLESS RUNNER. *Journal.Stekom.Ac.Id*, *15*(2), 254–266. http://journal.stekom.ac.id/index.php/pixel/article/view/770

Septiyani, A. (2022). *Indonesia Jadi Negara Kedua Pengguna Internet yang Banyak Main Game*. https://games.grid.id/read/153263015/indonesia-jadi-negara-kedua-pengguna-internet-yang-banyak-main-game

Sugiarto, H. (2022). PENERAPAN METODE GAME DEVELOPMENT LIFE CYCLE PADA APLIKASI GAME TEBAK NAMA PAHLAWAN NASIONAL BERBASIS ANDROID. *Ejournal.Pelitaindonesia.Ac.Id*, *6*(1), 1–7. https://www.ejournal.pelitaindonesia.ac.id/ojs32/index.php/JOISIE/article/view/1659

Tan, W. (2021). Analisis Performa Prototype Game Pada Platform Android. *Jurnal.Polibatam.Ac.Id*, *15*(2), 254–266. https://jurnal.polibatam.ac.id/index.php/JAMN/article/view/3781