Terbit online pada laman web jurnal: **http://journal.isas.or.id/index.php/JACOST**

JOURNAL OF APPLIED COMPUTER SCIENCE AND TECHNOLOGY (JACOST)

Vol. 1 No. 1 (2020) 1 - 6

Pengembangan Game Molemash di Android Menggunakan Metode Rapid Application Development

Agus Cahyo Nugroho
Sistem Informasi, Fakultas Ilmu Komputer, Unika Soegijapranata Semarang
agus.nugroho@unika.ac.id

Abstract

As we know the game is one of the interesting applications found in gadgets. Many people who play games use their gadgets. The fans of this game come from various age segments. Although the majority is dominated by children and adolescents. Molemash is one of the popular games since time immemorial. This Molemash game is played by touching certain characters that appear on the gadget's screen. The aim of developing this Molemash game is as a media to entertain children during the Covid-19 pandemic. Another goal is to introduce programming to today's parents. The development method used is Rapid Application Development (RAD). One application that supports the RAD method is MIT App Inventor 2. MIT App Inventor 2 is one of the tools that supports the RAD method because it implements block programming so that it allows efficiency in terms of program lines that must be written. One block at MIT App Inventor 2 represents dozens or even hundreds of program lines. Users only need to arrange the blocks according to the desired logic flow. Therefore MIT App Inventor 2 is suitable for rapid software development. The result of developing this Molemash game is that children get alternative entertainment during the Covid-19 pandemic. Whereas in terms of parents get new knowledge related to game development on Android.

Keywords: game, molemash, android, mit app inventor 2, RAD

Abstrak

Seperti kita ketahui game adalah salah satu aplikasi menarik yang terdapat di gadget. Banyak orang yg bermain game menggunakan gadget mereka. Para penggemar game ini berasal dari berbagai segmen usia. Meskipun mayoritas di dominasi oleh para anak dan remaja. Molemash adalah salah satu game yang populer sejak dulu kala. Game Molemash ini dimainkan dengan cara menyentuh karakter tertentu yang muncul di layar gadget. Tujuan pengembangan game Molemash ini adalah sebagai media penghibur anak selama masa pandemi Covid-19. Tujuan lain adalah untuk memperkenalkan pemrograman ke orang tua masa kini. Metode pengembangan yang digunakan adalah *Rapid Application Development (RAD)*. Salah satu aplikasi yang mendukung metode RAD adalah *MIT App Inventor 2*. *MIT App Inventor 2* adalah salah satu *tools* yang mendukung metode RAD karena menerapkan *block programming* sehingga memungkinkan efisiensi dalam hal baris program yang harus ditulis. Satu *block* di *MIT App Inventor 2* mewakili puluhan atau bahkan ratusan baris program. Pengguna hanya perlu menyusun *block-block* tersebut sesuai alur logika yang diinginkan. Oleh karena itu *MIT App Inventor 2* cocok digunakan untuk pengembangan perangkat lunak secara cepat. Hasil dari pengembangan game Molemash ini adalah anak mendapatkan alternatif hiburan selama masa pandemi Covid-19. Sedangkan dari sisi orang tua mendapatkan pengetahuan baru terkait pengembangan game di Android.

Kata kunci: game, molemash, android, mit app inventor 2, RAD

1. Pendahuluan

Beberapa bulan terakhir ini memaksa semua orang untuk beradaptasi dengan berbagai macam kebiasaan baru. Mulai dari menggunakan masker, mencuci tangan dengan sabun setiap kali setelah keluar rumah, menggunakan hand sanitizer hingga bekerja dan belajar di rumah. Awalnya setiap orang antusias karena bisa mengerjakan segala sesuatu dari rumah. Namun lama kelamaan pasti timbul bosan. Terutama bagi anak-anak yang mau tidak mau harus belajar di rumah dan tidak bisa bermain di luar rumah bebas seperti biasanya. Permasalahan tersebut tentu harus dicari solusinya. Solusi supaya anak-anak dapat terhibur meski harus tinggal di rumah. Selain itu, orang tua juga tetap mengasah keterampilan dengan belajar pemrograman tingkat dasar sebagai pengusir kejenuhan berada di rumah saja.

Salah satu tools pemrograman Android yg tepat untuk pemula adalah *MIT App Inventor 2*. Pemrograman dengan *MIT App Inventor 2* dilakukan dengan cara menyusun kepingan-kepingan block programming sesuai dengan logika kita. *MIT App Inventor 2* memungkinkan pengguna untuk beralih dari pekerjaan mereka dari laboratorium komputer konvensional ke

kehidupan sehari-hari dan komunitas mereka. Transisi ini memiliki dampak kuat pada apa yang pengguna ciptakan, solusi apa yang mereka berikan untuk menyelesaikan masalah dan bagaimana mereka mengembangkan diri mereka sebagai generasi digital. Hal ini memungkinkan pengguna untuk mengubah persepsi mereka tentang diri mereka sendiri dari individu yang "tahu cara membuat kode" menjadi bagian dari beberapa komunitas yang diberdayakan untuk memiliki solusi nyata dalam kehidupan mereka dan orang lain. *MIT App Inventor 2* telah mengubah pendidikan komputasi dari fokus pada teori menjadi fokus pada praktik, *MIT App Inventor 2* telah berhasil mengkonseptualisasikan pendidikan komputasi melalui lensa aksi komputasi dan mendukung pengguna untuk terlibat dalam komunitas yang lebih luas dari pembuat konten yang diberdayakan secara digital [1].

Pengembangan aplikasi *Android* dari awal mungkin tidak cocok atau efisien untuk semua kasus, terutama untuk proses pembuatan prototipe di mana para perancang seharusnya lebih fokus pada konsep dan desain aplikasi mobile [2]. Selain itu, para desainer dapat mencari solusi yang lebih sederhana dan lebih cepat untuk mengembangkan aplikasi mobile, terutama untuk semua orang tanpa keahlian pemrograman khusus [3]. Untuk alasan itu, kadang-kadang lebih disukai menggunakan editor visual untuk membuat prototipe aplikasi rumah pintar sehingga perancang dapat mengembangkan sketsa dan mengimplementasikan konsep yang dimaksud dengan lebih mudah. Salah satu aplikasi yang paling umum digunakan untuk tujuan tersebut adalah *MIT App Inventor 2*.

Saat ini, ada kecenderungan umum untuk mencari solusi yang lebih sederhana dan lebih cepat yang dapat digunakan dalam proses pengembangan perangkat lunak. Pada saat yang sama dua tren lain dapat diamati - peningkatan popularitas aplikasi seluler yang cepat dan memperkenalkan konsep-konsep TI kepada pemula [4]. Inilah sebabnya mengapa App Inventor 2 adalah salah satu alat yang diminati oleh komunitas pengembangan perangkat lunak.

MIT App Inventor 2 menggunakan keunggulan dari pemrograman visual komponen, di mana orang dapat menarik dan melepaskan komponen visual, dan kemudian memberikan perilaku yang dapat diprogram ke blok logika untuk mengembangkan aplikasi seluler dengan mudah. Oleh karena itu, *MIT App Inventor 2* yang digunakan untuk pembuatan prototipe aplikasi sangat efisien karena *MIT App Inventor 2* mudah digunakan dan intuitif untuk pemrograman aplikasi [5].

Penelitian ini diadakan dengan tujuan mengenalkan pemrograman ke orang tua sehingga tertarik untuk belajar lebih lanjut dan berkontribusi terhadap keluarga, masyarakat dan lingkungan sekitarnya.

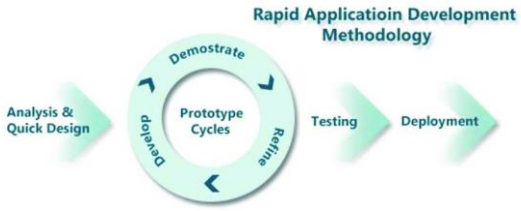
Sehingga di akhir penelitian ini bisa disimpulkan apakah *MIT App Inventor 2* merupakan tools yang tepat dalam mendukung implementasi RAD.

2. Metode Penelitian

2.1. *Rapid Application Development (RAD)*

Metode pengembangan perangkat lunak ini merancang desain siklus hidup perangkat lunak dengan menyediakan pengembangan secara cepat dan berkualitas tinggi. Efisiensi waktu adalah keunggulan RAD dibandingkan dengan metode pengembangan perangkat lunak lain dimana membutuhkan proses lebih lama. Pengembangan perangkat lunak menggunakan RAD adalah kesempatan bagus yang dapat dimanfaatkan dalam proses pengembangan perangkat lunak [6].

Metode ini kurang fokus pada perencanaan dan lebih fokus pada pengembangan. Oleh sebab itu, beberapa siklus pengembangan dapat berjalan secara bersamaan. Setiap siklus mempunyai dua fase pengembangan dan pengujian, yang dikenal dengan modul. Komentar dan masukan dari pemilik proyek diterima setelah setiap unit selesai. Metodologi ini cocok untuk usaha kecil, menengah dan besar, tetapi harus dipastikan bahwa proyek tersebut telah dibagi menjadi beberapa unit seperti yang ditunjukkan pada Gambar 1 [6].



Gambar 1. Metode RAD

2.2. Metode Riset

Metode penelitian yang digunakan terdiri dari 5 tahap, yaitu: analisis kebutuhan dan masalah, studi literatur dan desain, pembuatan aplikasi, implementasi, hasil dan kesimpulan.

2.2.1. Analisis Kebutuhan dan Masalah

Pada tahap ini peneliti melakukan pengamatan langsung ke lokasi penelitian. Peneliti melakukan wawancara dengan beberapa orang tua siswa yang sekolahnya memberlakukan kebijakan belajar di rumah.

2.2.2. Studi Literatur dan Desain

Setelah menganalisis kebutuhan dan masalah, langkah selanjutnya adalah melakukan studi literatur tentang penelitian serupa yang telah dilakukan sebelumnya, menganalisis saran dalam penelitian sebelumnya, memilih yang sesuai dan mengimplementasikannya dalam penelitian ini.

Penelitian dilanjutkan dengan proses desain baik desain antarmuka dan desain aliran logika game. Desain antarmuka menggambarkan berbagai bentuk yang digunakan dalam game, komponen-komponen karakter dan tampilan informasi apakah suatu proses berhasil atau tidak. Desain aliran logika game digunakan untuk

menjelaskan jalannya game dari awal hingga akhir. Alur logika game ini dapat diilustrasikan menggunakan *Flowchart*.

2.2.3. Pengembangan Aplikasi

User Interface adalah dasar dari pengembangan game sehingga setelah desain *User Interface* selesai, proses selanjutnya dilanjutkan dengan membuat *block programming*. Game tersebut dikembangkan menggunakan *MIT App Inventor 2* yang mendukung metode Pengembangan Aplikasi Cepat (RAD).

2.2.4. Implementasi

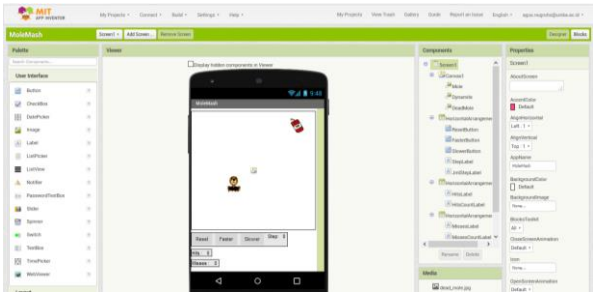
Setelah game dikembangkan, langkah selanjutnya adalah mengimplementasikan game pada objek penelitian. Hal ini dilakukan agar game tersebut dapat menjadi alternatif hiburan bagi anak di rumah.

2.2.5. Hasil dan Pembahasan

Hasil dan diskusi adalah tahap akhir dari penelitian ini di mana penulis akan menyimpulkan hasil pengamatan setelah game diimplementasikan. Pada tahap ini pengamatan juga akan dilakukan dan masukan kira-kira apa yang dapat ditambahkan untuk penelitian lebih lanjut.

2.3. MIT App Inventor 2

MIT App Inventor 2 adalah platform pengembangan online yang bisa digunakan oleh siapa saja untuk memecahkan masalah dunia nyata. *MIT App Inventor 2* menyediakan editor ramah pengguna berbasis web dengan fitur "What you see is what you get" (WYSIWYG) untuk mengembangkan aplikasi mobile berbasis sistem operasi Android dan iOS. Tampilan halaman utama *MIT App Inventor 2* bisa kita lihat di Gambar 2.

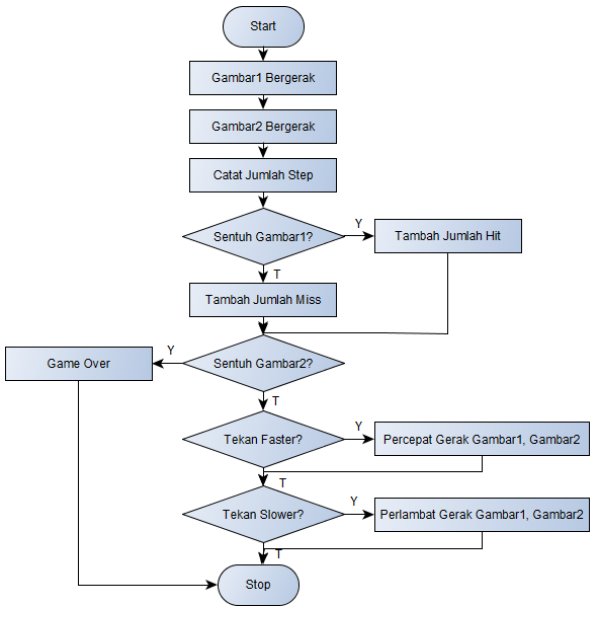


Gambar 2. Tampilan Halaman Utama *MIT App Inventor 2*

MIT App Inventor 2 menggunakan bahasa pemrograman berbasis blok yang dibangun di *Google Blockly* [7] dan terinspirasi oleh bahasa seperti *StarLogo TNG* [8] dan *Scratch* [9], memungkinkan siapa pun dengan berbagai tingkat pengetahuan untuk membangun aplikasi seluler untuk memenuhi kebutuhan dan menyelesaikan masalah di masyarakat. Bahasa pemrograman berbasis blok di *MIT App Inventor 2* dapat kita lihat pada Gambar 3. Hingga saat ini, 6,8 juta orang di lebih dari 190 negara telah menggunakan *MIT App Inventor 2* untuk membangun lebih dari 24 juta aplikasi. *MIT App Inventor 2* menawarkan antarmuka dalam lebih dari selusin bahasa. Orang-orang di seluruh dunia menggunakan *MIT App Inventor 2* untuk memberikan solusi seluler untuk masalah nyata dalam keluarga, komunitas, dan dunia mereka. Platform ini juga telah diadaptasi untuk melayani kebutuhan populasi yang lebih spesifik, seperti membangun aplikasi untuk responden darurat / pertama dan robotika [10].

3. Hasil dan Pembahasan

Game ini dimulai dengan bergeraknya Gambar1 dan Gambar2. Kedua gambar ini merupakan komponen *ImageSprite* dari *MIT App Inventor 2*. Gambar1 kita ganti dengan gambar Mole. Sementara Gambar2 kita ganti dengan gambar TNT. Saat game dimulai ada counter yang mencatat jumlah step atau pergerakan dari Gambar1 dan Gambar2. Selanjutnya jika pengguna berhasil menyentuh Gambar1 yang bergerak maka label jumlah hit akan bertambah, namun jika pengguna gagal menyentuh Gambar1 yang bergerak dan mengenai Canvas permainan maka label jumlah miss yang akan bertambah. Sedangkan jika pengguna menyentuh Gambar2 yang bergerak maka itu berarti game berakhir atau game over. Pada menu permainan terdapat juga dua tombol yaitu: faster dan slower. Tombol faster fungsinya mempercepat pergerakan Gambar1 dan Gambar2. Sebaliknya tombol slower digunakan untuk memperlambat pergerakan Gambar1 dan Gambar2. Tombol terakhir yaitu: reset digunakan untuk memulai kembali game dari awal dan mengembalikan jumlah hit dan miss ke angka 0. Alur jalannya game di atas bisa kita lihat pada Gambar 3.



Gambar 3. Flowchart Game Molemash

Langkah pertama dalam pembuatan game ini adalah desain user interface terlebih dahulu. Pertama kita tarik dan letakan *Canvas* dari *Palette Drawing and Animation* ke *Viewer*. *Canvas* ini adalah tempat dimana kita menaruh *ImageSprite* yaitu: karakter 1 Mole, karakter 2

Dynamite dan karakter 3 Dead Mole. Karakter pertama Mole adalah gambar yang akan bergerak sepanjang game berjalan dan harus kita sentuh untuk mendapatkan poin di jumlah hit. Karakter Mole ini bisa kita lihat pada Gambar 4.

Gambar 4. Karakter 1 Mole

Karakter kedua Dinamit adalah gambar yang juga akan bergerak sepanjang game ini. Namun apabila kita menyentuh Dynamite ini maka game akan berakhir atau game over. Karakter Dynamite ini bisa kita lihat pada Gambar 5.

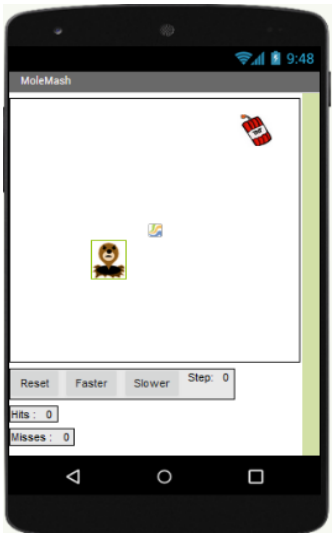
Gambar 5. Karakter 2 Dynamite

Karakter ketiga Dead Mole adalah gambar yang akan muncul setelah kita menyentuh karakter kedua yaitu: Dynamite. Kemunculan karakter ketiga ini menandakan kematian karakter pertama Mole akibat terkena karakter kedua Dynamite. Karakter ketiga ini bisa kita lihat pada Gambar 6.

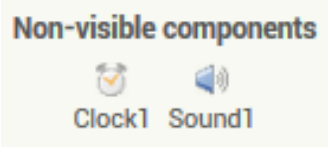
Gambar 6. Karakter 3 Dead Mole

Secara keseluruhan tampilan user interface lengkap bisa kita lihat pada Gambar 7. Pada user interface tersebut terdapat berbagai macam komponen antara lain: tiga buah *ImageSprite* pada *Canvas*, tiga buah tombol yaitu: Reset, Faster, Slower dan dua buah label yaitu: StepLabel dan JumlahStepLabel yang kita atur menggunakan layout *HorizontalArrangement*, HitsLabel beserta HitsCountLabel yang kita atur juga menggunakan layout *HorizontalArrangement* dan paling bawah MissesLabel, MissesCountLabel yang kita taruh juga di dalam layout *HorizontalArrangement*.

Selain berbagai komponen user interface yang kita gunakan di atas masih ada dua buah komponen Non-visible lagi yaitu: *Clock* dan *Sound*. Komponen *Clock* kita gunakan untuk menggerakan *ImageSprite* Gambar1 dan Gambar2 tiap detik. Sementara komponen *Sound* kita gunakan supaya apabila *ImageSprite* Gambar1 berhasil disentuh oleh pengguna maka gadget akan bergetar. Tampilan komponen Non-visible ini bisa kita lihat di Gambar 8.

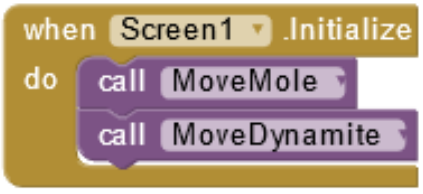


Gambar 7. Tampilan User Interface Lengkap



Gambar 8. Tampilan Non Visible Components

Sekarang kita bahas block programming dari game Molemash tersebut. Block programming di bawah ini kita peroleh melalui komponen Screen1. Jadi saat game mulai berjalan maka Block Programming tersebut akan memanggil prosedur MoveMole dan MoveDynamite. Kedua prosedur tersebut akan mulai menggerakan *ImageSprite* Gambar1 Mole dan *ImageSprite* Gambar2 Dynamite. Tampilan Block Screen 1 *Initialize* bisa kita lihat pada Gambar 9.



Gambar 9. Tampilan Block Screen1 *Initialize*

Pada prosedur MoveMole dan MoveDynamite kita menggunakan fungsi Math *RandomInteger* supaya *ImageSprite* Gambar1 dan Gambar2 muncul di tempat yang berbeda-beda setiap detik. Block *Random Integer* ini bisa kita lihat pada Gambar 10.

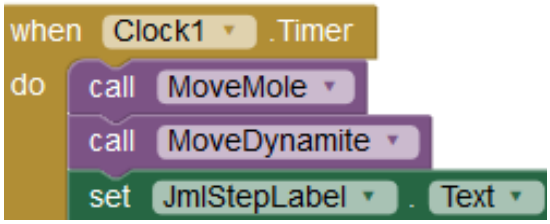
Gambar 10. Tampilan Block *Random Integer*

Kemudian kita gunakan Block Canvas1 *Touched* untuk mendeteksi saat pengguna menyentuh *ImageSprite* atau *Canvas*. Jika yang disentuh *ImageSprite* Gambar1 maka jumlah hits akan bertambah satu sementara di luar

sentuhan terhadap *ImageSprite* Gambar1 maka jumlah misses yang akan bertambah. Jumlah hits dan misses ini akan kembali ke 0 lagi saat pengguna menekan tombol Reset. Tampilan Block Canvas1 *Touched* bisa kita lihat pada Gambar 11.

Gambar 11. Tampilan Block Canvas1 *Touched*

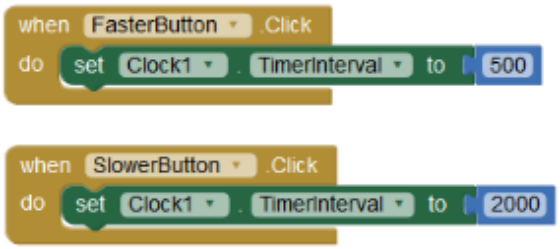
Berikutnya komponen Non-visible yaitu: *Clock* yang digunakan untuk menggerakan posisi *ImageSprite* Gambar1 dan Gambar2 tiap detik secara random. Hal ini dimungkinkan melalui pemanggilan prosedur MoveMole dan MoveDynamite. Selain itu game juga menghitung jumlah step yang sudah dilakukan oleh *ImageSprite* Gambar1 dan Gambar2.



Gambar 12. Tampilan Block Clock1 *Timer*

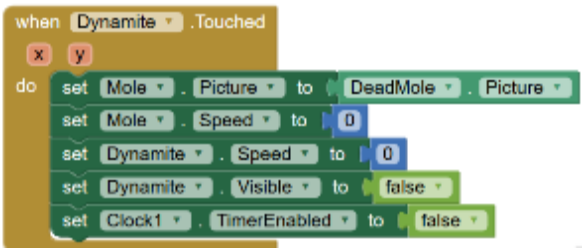
Pada game juga terdapat tombol Faster dan Slower. Tombol Faster berguna untuk mempercepat pergerakan *ImageSprite* Gambar1 Mole dan *ImageSprite* Gambar2 Dynamite. Tombol Slower berguna untuk memperlambat pergerakan *ImageSprite* Gambar1 Mole dan ImageSprite Gambar2 Dynamite. Hal ini dilakukan dengan mengatur *TimerInterval* pada komponen Non-visible Clock1. Pada tombol Faster kita atur *TimerInterval* menjadi 500 miliseconds atau setengah detik. Hal ini berarti *ImageSprite* Gambar1 Mole dan *ImageSprite* Gambar2 Dynamite akan berpindah posisi setiap 500 miliseconds atau setengah detik. Sedangkan pada tombol Slower kita atur *TimerInterval* menjadi 2000 miliseconds atau 2 detik. Hal ini berarti *ImageSprite* Gambar1 Mole dan *ImageSprite* Gambar2 Dynamite akan berpindah posisi setiap 2000 miliseconds atau dua detik. Pengaturan *TimerInterval* ini bisa kita lihat pada Gambar 13.

Kita mempunyai *ImageSprite* Gambar1 Mole dan *ImageSprite* Gambar2 Dynamite. Jika pengguna menyentuh *ImageSprite* Gambar2 Dynamite yang terjadi adalah game akan selesai atau game over.



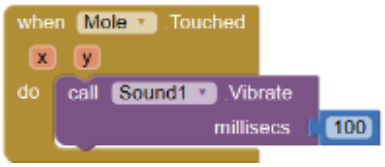
Gambar 13. Tampilan Block Faster and Slower Button

Saat pengguna menyentuh *ImageSprite* Gambar2 yang dilakukan oleh Block Dynamite *Touched* adalah pertama mengubah *ImageSprite* Gambar1 Mole menjadi gambar DeadMole. Kemudian mengubah kecepatan *ImageSprite* Gambar1 Mole dan *ImageSprite* Gambar2 Dynamite menjadi 0. Hal ini dilakukan supaya kedua ImageSprite Gambar1 dan Gambar2 tersebut berhenti bergerak. Selanjutnya ubah *ImageSprite* Gambar2 Dynamite *visible* False. Hal ini kita lakukan supaya *ImageSprite* Gambar2 Dynamite tidak terlihat di layar. Terakhir kita hentikan semua pergerakan *ImageSprite* Gambar1 dan Gambar2 dengan cara mengubah komponen Non-visible Clock1 *TimerEnabled* menjadi False. Hal ini juga kita lakukan supaya komponen Non-visible Clock1 menjadi tidak aktif. Block lengkapnya bisa kita lihat pada Gambar 14.



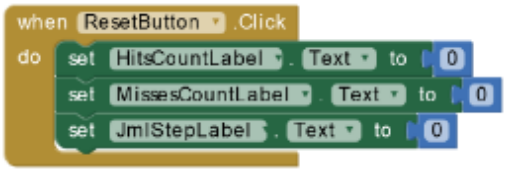
Gambar 14. Tampilan Block Dynamite *Touched*

Selanjutnya adalah Block yang mengatur saat pengguna menyentuh *ImageSprite* Gambar1 Mole maka gadget akan bergetar. Hal ini bisa kita lakukan dengan memanggil komponen Non-visible Sound1 *Vibrate* dan memberikan masukan bergetar selama 100 miliseconds. Block ini bisa kita lihat lebih detil pada Gambar 15.



Gambar 15. Tampilan Block Mole *Touched*

Terakhir adalah Block tombol Reset. Pada block ini yang kita lakukan adalah mengubah label jumlah hit kembali ke 0. Kita ubah juga label jumlah miss kembali ke 0. Selain itu kita ubah juga label jumlah step kembali ke 0. Tampilan lebih detil dari block ini bisa kita lihat pada Gambar 16.



Gambar 16. Tampilan Block Reset Button

4. Kesimpulan

Permasalahan terkait kurangnya alternatif hiburan anak selama pandemi Covid 19 bisa diatasi dengan adanya game ini. Game Molemash yang ringan dan mudah untuk dimainkan cocok digunakan sebagai alternatif hiburan selama pandemi Covid 19. Selain itu orang tua juga bisa mengasah keterampilan berpikir logis yang dapat langsung diimplementasikan dan bermanfaat bagi keluarga. Orang tua bisa menggunakan block programming menggunakan *MIT App Inventor 2* yang terbukti mudah digunakan dan diimplementasikan oleh pemula di seluruh dunia. Hal ini bisa kita lihat dari hasil survey terhadap kesan peserta selama menggunakan *MIT App Inventor 2* untuk membuat game pertama mereka pada Gambar 17.

Kesan Mengikuti Kursus Online
"Cara Mudah Membuat Aplikasi Mobile di Android"
Seru, Asik, Bermanfaat
Tidak menyangka membuat aplikasi ada versi mudahnya
Menambah pengalaman
Dosen menjelaskan dengan sabar langkah demi langkah
Menambah wawasan dan bisa untuk dipraktekkan sendiri untuk mengisi waktu di rumah saja
Bangga bisa merasakan bangku kuliah meski melalui online

Gambar 17. Kesan dari Peserta

MIT App Inventor 2 dengan block programmingnya terbukti tools yang tepat dalam mendukung implementasi RAD karena meringkas baris-baris program yang panjang ke dalam block programming sehingga waktu pengembangan menjadi lebih singkat dan cepat. Selain itu hadirnya *MIT AI2 Companion* juga mendukung pengembangan perangkat lunak secara cepat karena pengguna bisa melakukan uji coba secara *live* dan *real time* di perangkat *Android* mereka. *MIT AI2 Companion* menghadirkan *live testing environment* sehingga pengguna bisa melakukan *debugging error* dan melihat hasilnya saat itu juga di *device* mereka. Hal ini tentunya semakin mempersingkat waktu pengembangan sehingga semakin menguatkan *MIT App Inventor 2* adalah salah satu *tools* yang tepat dalam mengimplementasikan RAD.

Pengembangan game di masa depan dapat dipertimbangkan fitur level dengan berbagai tingkat kesulitan yang bertingkat. Sehingga pengguna akan semakin tertantang dalam memainkan game ini. Kemudian bisa ditambahkan efek suara yang lebih menarik sehingga pengguna akan semakin betah dalam memainkan. Demikian juga karakter dalam game juga bisa ditambah lagi supaya lebih bervariasi.

Daftar Rujukan

[1] Kong, S. C., & Abelson, H. (Eds.). (2019). Computational Thinking Education. Springer.
[2] Kang, H., Cho, J., & Kim, H. (2015). Application study on Android application prototyping method using App inventor. *Indian Journal of Science and Technology*, *8*(19), 1–5.
[3] Radoslaw, K., Turczynski, L., & Zyla, K. (2016). Comparison of App Inventor 2 and Java in creating personal applications for Android on example of a notepad. *Advances in Science and Technology Research Journal*, *10*(31), 247–254.
[4] Kowalczyk, R., Turczyński, Ł., & Żyła, K. (2016). Comparison of App Inventor 2 and Java in creating personal applications for Android on example of a notepad. *Advances in Science and Technology. Research Journal*, *10*(31).
[5] Kang, H., Cho, J., & Kim, H. (2015). Application study on android application prototyping method using app inventor. *Indian Journal of Science and Technology*, *8*(18), 1.
[6] Saeed, S., Jhanjhi, N. Z., Naqvi, M., & Humayun, M. (2019). Analysis of Software Development Methodologies. International Journal of Computing and Digital Systems, 8(5), 446-460.
[7] Fraser, N. (2013). Blockly: A visual programming editor. URL: https://code. google. com/p/blockly.
[8] Begel, A., & Klopfer, E. (2007). Starlogo TNG: An introduction to game development. Journal of E-Learning, 53, 146.
[9] Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. B. (2009). Scratch: Programming for all. Commun. Acm, 52(11), 60-67.
[10] Papadakis, S., & Orfanakis, V. (2016, November). The combined use of Lego Mindstorms NXT and App Inventor for teaching novice programmers. In International Conference EduRobotics 2016 (pp. 193-204). Springer, Cham.