# RANCANG BANGUN APLIKASI GAME DUCK HUNTER BERBASIS ANDROID

# SKRIPSI


**Disusun Oleh :**

**DAVID JOAN PERMANA**

**06.12.507**



**JURUSAN TEKNIK ELEKTRO S-1**

**KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA**

**FAKULTAS TEKNOLOGI INDUSTRI**

**INSTITUT TEKNOLOGI NASIONAL MALANG**


**2012**

**LEMBAR PERSETUJUAN**

# PENGEMBANGAN APLIKASI GAME DUCK HUNTER BERBASIS ANDROID

## SKRIPSI

*Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Komputer dan Informatika Strata Satu (S-1)*


**Disusun oleh :**

**DAVID JOAN PERMANA**

**06.12.507**


**Mengetahui,**

**Ketua Jurusan Teknik Elektro S-1**

**Ir. Yusuf Ismail Nakhoda, MT**

**NIP. Y. 101 880 0189**

**Diperiksa dan Disetujui**

| Dosen Pembimbing I | Dosen Pembimbing II |
|---|---|
| **Sandy Nataly Mantja, S.KOM.** | **Ibrahim Ashari, ST, MT.** |
| **NIP.P 103.0800.418** | **NIP.Y 101.8700.151** |

**JURUSAN TEKNIK ELEKTRO S-1**
**KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA**
**FAKULTAS TEKNOLOGI INDUSTRI**
**INSTITUT TEKNOLOGI NASIONAL MALANG**
**2011**

# PENGEMBANGAN APLIKASI GAME DUCK HUNTER BERBASIS ANDROID

**David Joan Permana**

Jurusan Teknik Elektro S-1, Konsentrasi Teknik Komputer dan Informatika
Fakultas Teknologi Industri, Institut Teknologi Nasional Malang
Jl. Raya Karanglo Km 2 Malamg
Email : davidjoan1988@gmail.com

Dosen Pembimbing : I. **Sandy Natali Mantja, SKom**
II. **M.Ibrahim Ashari, ST, MT**



***Abstraksi***

*Dalam sebuah handphone pada saat ini terasa tidak lengkap rasanya bila tidak terdapat sebuah permainan. Sebuah permainan menjadi salah satu pelengkap di dalam sebuah handphone, baik permainan yang bersifat menghibur maupun permainan yang membutuhkan pikiran untuk memainkannya. Permainan atau game Duck Hunter adalah salah satu game yang bergenre shooting yang bersifat menghibur, jenis permainan ini mempunyai level yang singkat, kontrol yang mudah, serta tingkat kesulitan yang bertambah dengan waktu yang singkat. permainan ini dirancang untuk memancing adrenalin permain serta tidak membutuhkan jalan cerita yang bagus dan kadang-kadang membutuhkan kelincahan dalam memegang kontrol dan membutuhkan waktu belajar yang relatif singkat sehingga permainan ini dapat menghibur para pemain dan dapat mengisi waktu kosong yang tidak pemikiran keras sehingga dapat dimainkan dengan santai tetapi tetap mempunyai daya tarik untuk dimainkan.*

*Kata kunci : game shooting, Duck Hunter*

# KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT, atas karunia yang telah dilimpahkanNya sehingga penulis dapat menyelesaikan tugas akhir dengan judul PENGEMBANGAN APLIKASI GAME DUCK HUNTER BERBASIS ANDROID.

Selanjutnya pada kesempatan ini penulis juga menyampaikan rasa terimakasih yang sebesar-besarnya kepada pihak–pihak yang telah banyak membantu penulis selama penyusunan tugas akhir, diantaranya :

1. Bapak Ir. Yusuf Ismail Nahkoda, MT selaku Ketua Jurusan Teknik Elektro S-1 ITN Malang.
2. Bapak Dr. Aryuanto Soetedjo, ST, MT selaku Sekertaris Jurusan Teknik Elektro S-1 ITN Malang dan pengusul serta penyedia ruang Skripsi.
3. Ibu Sandy Nataly Mantja, S.KOM selaku Dosen Pembimbing I
4. Bapak Ibrahim Ashari, ST, MT selaku Dosen Pembimbing II
5. Bapak Ahmad Faisol, ST selaku Dosen Wali.
6. Kedua orang tua yang telah memberikan dukungan untuk selalu berdoa, berusaha dan nasehat yang telah diberikan sampai saat ini.
7. Seluruh dosen dan pegawai ITN Kampus 2 Malang.
8. Semua teman-teman ITN Kampus 2 Malang dari angkatan 2006-2010
9. Semua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini yang tidak bisa penulis sebutkan satu persatu..

Tentunya laporan tugas akhir ini masih memiliki banyak kekurangan dan kelemahan dalam penyusunannya. Oleh karena itu penulis mengharapkan kritik dan saran yang membangun demi kesempurnaan penyusunan tugas akhir ini. Besar harapan penulis laporan tugas akhir ini dapat bermanfaat bagi semua pihak.

Malang, Agustus 2012

Penulis

# DAFTAR ISI

# DAFTAR TABEL

# DAFTAR GAMBAR

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Seiring perkembangan teknologi pada saat ini telah banyak melahirkan permainan. Permainan saat ini telah banyak dimainkan oleh banyak orang dari usia muda sampai tua. Ada banyak genre dalam game, salah satunya adalah *game Shooting*. Game Shooting adalah video game genre ini sangat memerlukan kecepatan refleks, koordinasi mata-tangan, juga timing, ciri-ciri dari game jenis ini adalah mempunyai level yang singkat, kontrol yang mudah, serta tingkat kesulitan yang bertambah dengan waktu yang singkat. Jenis permainan ini dirancang untuk memancing adrenalin permain serta tidak membutuhkan jalan cerita yang bagus. Permainan ini kadang-kadang membutuhkan kelincahan dalam memegang kontrol dan membutuhkan waktu belajar yang relatif singkat.. Pada masa jaya konsol NES, ada sebuah game tembak - tembakan yang cukup unik pada saat itu yaitu game Duck Hunt. Game yang berjudul Duck Hunt tersebut memperlihatkan beberapa bebek yang terbang berlintasan diatas pepohonan, dan harus menembak bebek tersebut menggunakan pistol infra merah yang sudah disediakan sepaket dengan gamenya. Interaksi yang unik ini merupakan inovasi yang benar - benar menarik minat para gamer untuk memainkannya. Terbukti dari hasil penjualan game Duck Hunt yang merupakan game dengan penjualan terbaik mencapai hampir 30 juta unit di seluruh dunia. Permainan ini dikembangkan dan diterbitkan oleh *Nintendo*, dan dirilis pada tahun 1984 di Negara jepang.

Berdasarkan penjelasan tersebut diatas, diperlukan pengembangan game android untuk membuat game bergenre shooting yang lebih menantang yaitu dengan adanya level di dalam game. Untuk itu penulis mencoba merealisasikannya pada Skripsi dengan judul "Rancang Bangung Aplikasi Game Duck Hunter Berbasis Android". Dengan aplikasi ini diharapkan dapat memberikan hiburan atau menghilangkan kejenuhan bagi pengguna handphone android. Pada sistem android terdapat game Duck Hunter yang di rilis pada tanggal 11 november 2011, permainan menembak yang bercerita tentang bebek-

Tugas anda di permainan ini adalah menembak bebek-bebek yang terbang dengan senjata yang anda miliki. Jika satu bebek lolos dari tembakan anda, maka nyawa anda akan hilang satu. Untuk mendapatkan skor tertinggi, ada baiknya anda menembak secara berturut-turut, dan jika hit rate anda 98% maka anda akan mendapatkan penghargaan berupa +1 nyawa. Karena game tersebut hanya mempunyai 5 level dan kurang menantang bagi penulis maka penulis ingin mengembangkan permainan ini dalam segi level dan tingkat kesulitan game tersebut dengan software Eclipse sebagai tempat pembuatan source code yang menggunakan bahasa pemograman android.

## 1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas, rumusan masalah yang dapat diambil dari pembuatan tugas akhir ini

1. Penggunaan bahasa pemograman android pada game duck hunter.
2. Bagaimana mengembangkan game *Duck Hunt* menggunakan bahasa pemograman android menggunakan software Eclipse sebagai tempat pembuatan source code aplikasi game.

## 1.3 Batasan Masalah

Batasan masalah yang dimasud agar pembahasan dapat dilakukan secara terarah dan tercapai sesuai dengan yang diharapkan, maka perlu ditetapkan batasan-batasan permasalahan yang akan di bahas, yaitu :

1. Pembuatan game menggunakan bahasa pemograman Android.
2. Menggunakan sebuah *Hand Phone* dengan sistem android.
3. Game *Duck Hunter* ini mempunyai 10 level senjata.
4. Tidak membahas game lain kecuali *game Duck Hunter* pada hand phone dengan sistem android.
5. Penggunaan diimplementasikan pada handphone dengan resolusi 240 x 320 khususnya Samsung Galaxy Young.

## 1.4 Tujuan Skripsi

Tujuan dan maksud dari penulisan laporan skripsi ini adalah :

1. Menghasilkan sebuah aplikasi game Duck Hunter berbasis android yang dapat di nikmati oleh para pemain.
2. Mengembangkan sebuah aplikasi game Duck Hunter yang lebih menarik dan menantang supaya para pemain tidak cepat bosan untuk memainkan game tersebut.

## 1.5 Metode Penelitian

Metologi penelitian yang digunakan pada skripsi ini adalah :

1. Studi Literatur

   Mempelajari buku-buku literature mengenai bahasa pemrograman yang digunakan, dan sumber ilmiyah lainnya seperti website, artikel dan dokumen text sebagai landasan yang ada hubungannya dengan permasalahan yang dijadikan objek penelitian.

2. Perancangan Sistem

   Dalam perancangan sistem ini penulis merancang sistem dengan menggunakan Eclipsce untuk mendesain sistem yang akan digunakan pada permainan tersebut.

3. Coding atau Pemrograman

   Tahap ini dilakukan implementasi hasil rancangan kedalam baris-baris kode program yang dapat dimengerti oleh mesin.

4. Implementasi dan Pengujian

   Aplikasi yang telah selesai diimplementasikan akan diuji coba, yaitu pengujian berdasarkan fungsional program dan dilakukan koreksi serta penyempurnaan program jika diperlukan

## 1.6 Sistematika Penulisan

Setelah dilakukan proses pelaksanaan dan pembuatan aplikasi game Ducj Huner berbasis android pada skripsi ini mulai dari studi literatur, perencanaan, pembuatan, pengujian dan perbaikan serta hasil – hasil yang didapat, maka untuk pembahasan selengkapnya diwujudkan dalam bentuk buku laporan skripsi dengan sistematika sebagai berikut :

**BAB I : PENDAHULUAN**

Dalam bab ini dibahas latar belakang tugas akhir yang dilaksanakan, maksud dan tujuan, batasan masalah, metologi penelitian, dan sistematik penelitian.

**BAB II : LANDASAN TEORI**

Pada bab ini meliputi literatur, dasar teori, serta referensi yang berguna sebagai acuan dan landasan bagi penulis dalam mengerjakan skripsi ini.

**BAB III : PERANCANGAN DAN ANALISA SISTEM**

Pada bab ini menjelaskan tentang perancangan umum maupun uraian lebih lanjut mengenai perancangan sistem dalam pembuatan perangkat lunak. Uraian perancangan aplikasi ini meliputi perancangan data mengenai input data serta output sistem, perancangan proses mengenai bagaimana sistem akan bekerja dengan proses-proses tertentu dan implementasi yang akan digunakan dalam pembuatan skripsi ini.

**BAB IV : PEMBUATAN DAN PENGUJIAN SISTEM**

Dalam bab ini penulis akan menganalisa pengujian dari perancangan dan pembuatan sistem yang telah dibuat secara keseluruhan.

**BAB V : KESIMPULAN DAN SARAN**

Pada bab ini berisikan kesimpulan yang dapat diambil berdasarkan hasil uraian pada bab-bab sebelumnya dan juga saran-saran serta masukan, untuk memperbaiki kelemahan sistem yang telah dibuat demi pengembangan dan penyempurnaan di waktu mendatang.

# BAB II
# LANDASAN TEORI

## 2.1 Sejarah Game

Game saat ini dimainkan oleh anak anak sampai tua, Game dalam artian bahasa indonesia adalah permainan. Game-game tradisional, seperti patok lele, petak umpet dan masih banyak game-game lainnya tetapi permainan seperti itu sudah sangat jarang sekali di temukan di kota-kota besar. Sekarang game yang sangat popular yang menjamur adalah game-game seperti game Online yang dapat merusak masa depan apabila tidak dikondisikan dengan benar oleh pemainnya.

Generasi game pertama muncul dari ATARI 2600 merupakan konsole game pertama yang sukses di mansanya, atari 2600 ini dirilis pada oktober 1977, setelah itu dikenal dengan nama VCS (Video Computer System), setelah sukses pada generasi pertama atari tidak menyerah sampai disitu saja untuk memajukan dunia game elektronik pada generasi kedua munculah atari 7800 konsol ini dirilis pada juni 1986, pada atari 7800 ini ada sedikit kemajuan dengan menambahkan joystik sehingga user dapat lebih mudah untuk memainkan konsole game ini namun pada masanya harga dari konsole game ini selangit yaitu $140. Lalu setelah Atari 7800 ada NES Nintendo entertaintment System inilah konsole permainan pertama kali yang menggunakan 8 bit, nintendo ini menghasilkan produk produk game yang lain dari pada yang lain salah satu game yang compatible dengan NES ini adalah super Mario bros.

Setelah NES laris manis dengan super marionya lalu muncul lagi konsole game Sega Mega Drive. Sega mega drive ini merupakan generasi ke tiga dari dunia game sega menggunakan 16 bit dan dirilis pada tahun 1988, kalau tadi nintendo dikenal dengan super mario brossnya sega juga tidak mau kalah dengan mengeluarkan game sonic the hedhog.

Pada generasi ke empat dari sejarah game produksi game semakin menjamur dan berkembang pesat salah satu contoh pada generasi ke empat ini lahirlah playstation merupakan salah satu game konsole terlaris dan menurut saya

terbaik di indonesia bahkan di dunia merupakan terobosan baru di dunia game dengan menggunakan 32 bit.

Generasi kelima playstation atau yang lebih akrab kita sebut PS semakin gencar melakukan produksi lainnya yang lebih mutakhir dengan mengeluarkan konsole game PS2 namun pada generasi ke lima ini sangat banyak sekali saingan saingan PS2 ,XBox, sega Saturn, Dreamcast, dari generasi kelima ini sudah mulai banyak perusahaan perusahaan yang memproduksi konsole game.

Generasi sekarang, pada tahun ini sudah banyak sekali konsole game menjamur di dunia PS juga masih terus mengembangkan sistemnya dengan mengeluarkan PS3, Xbox juga mengeluarkan XBOX 360, lalu nintendo juga mengeluarkan WII.

Generasi sekarang ini sedang dikembangkan game yang menggunakan kacamata khusus sehingga user dapat menikmati sebuah game seperti seolah-olah nyata.

## 2.2. Jenis-jenis Game

Pada saat ini kita bias temukan macam-macam game seperti Arcade Game, PC Game, Console Game (seperti Playstation 2, Playstation 3, XBOX 360, dan Nintendo), Handheld games, dan Mobile games. Dari banyaknya game yang ada pada saat ini, berdasarkan genre game dapat diklasifikasikan menjadi beberapa jenis.

### 2.2.1 Aksi – Shooting

Aksi – Shooting (tembak-tembakan , atau hajar-hajaran bisa juga tusuk-tusukan, tergantung cerita dan tokoh di dalamnya), video game jenis ini sangat memerlukan kecepatan refleks, koordinasi mata-tangan, juga timing, inti dari game jenis ini adalah tembak, tembak dan tembak. Termasuk didalam-nya :

a.First person shooting (FPS) seperti Counter Strike dan Call of Duty

b.Drive n' shoot, menggunakan unsur simulasi kendaraan tetapi tetap dengan tujuan utama menembak dan menghancurkan lawan, contoh : Spy Hunter, Rock and Roll Racing, Road Rash.

c. Shoot em' up, seperti Raiden, 1942, dan gradius.

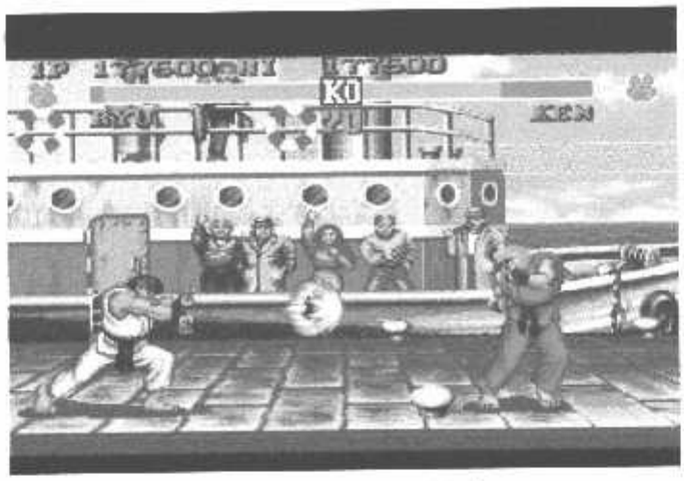
d. Beat 'em up (tonjok hajar) seperti Double Dragon dan Final Fight, lalu **hack and slash** (tusuk tebas) seperti Shinobi dan Legend of Kage.

e. Light gun shooting, yang menggunakan alat yang umumnya berbentuk seperti senjata, seperti Virtua Cop dan Time Crisis.

Gambar 2.1 Game Shooting

### 2.2.2 Fighting ( pertarungan )

Fighting ( pertarungan ) Ada yang mengelompokan video game fighting di bagian Aksi, namun penulis berpendapat berbeda, jenis ini memang memerlukan kecepatan refleks dan koordinasi mata-tangan, tetapi inti dari game ini adalah penguasaan jurus (hafal caranya dan lancar mengeksekusinya), pengenalan karakter dan timing sangatlah penting, o iya, *combo*-pun menjadi esensial untuk mengalahkan lawan secepat mungkin. Dan berbeda seperti game Aksi pada umumnya yang umumnya hanya melawan Artificial Intellegence atau istilah umumnya *melawan komputer* saja, pemain jenis *fighting game* ini baru teruji kemampuan sesungguhnya dengan melawan pemain lainnya. Seri Street Fighter, Tekken, Mortal Kombat, Soul Calibur dan King of Fighter adalah contohnya.




Gambar 2.2 Game Fighting

### 2.2.3 Aksi – Petualangan

Memasuki gua bawah tanah, melompati bebatuan di antara lahar, bergelayutan dari pohon satu ke pohon lain, bergulat dengan ular sambil mencari kunci untuk membuka pintu kuil legendaris, atau sekedar mencari telepon umum untuk mendapatkan misi berikutnya, itulah beberapa dari banyak hal yang karakter pemain harus lakukan dan lalui dalam video game jenis ini. Menurut penulis, game jenis ini sudah berkembang jauh hingga menjadi genre campuran *action beat-em up* juga, dan sekarang, di tahun 2000 an, jenis ini cenderung untuk memiliki visual 3D dan sudut pandang orang ke-tiga. Tomb Rider, Grand Theft Auto dan Prince of Persia termasuk didalamnya.

Gambar 2.3 Game aksi petualangan

### 2.2.4 Petualangan

Petualangan, Bedanya dengan jenis video game aksi-petualangan, refleks dan kelihaian pemain dalam bergerak, berlari, melompat hingga memecut atau menembak tidak diperlukan di sini. Video Game murni petualangan lebih menekankan pada jalan cerita dan kemampuan berpikir pemain dalam menganalisa tempat secara visual, memecahkan teka-teki maupun menyimpulkan rangkaian peristiwa dan percakapan karakter hingga penggunaan benda-benda tepat pada tempat yang tepat. Termasuk didalamnya:

a. Petualangan dengan teks atau sistem tunjuk dan klik, contoh: Kings Quest, Space Quest, Heroes Quest, Monkey Island, Sam and Max,
b. Novel atau film interaktif, seperti game "dating" yang banyak beredar di jepang, Dragons Lair dan Night Trap.

Gambar 2.4 Game petualangan

### 2.2.5 Simulasi

Simulasi, Konstruksi dan manajemen. Video Game jenis ini seringkali menggambarkan dunia di dalamnya sedekat mungkin dengan dunia nyata dan memperhatikan dengan detil berbagai faktor. Dari mencari jodoh dan pekerjaan, membangun rumah, gedung hingga kota, mengatur pajak dan dana kota hingga keputusan memecat atau menambah karyawan. Dunia kehidupan rumah tangga sampai bisnis membangun konglomerasi, dari jualan limun pinggir jalan hingga membangun laboratorium cloning. Video Game jenis ini membuat pemain harus

berpikir untuk mendirikan, membangun dan mengatasi masalah dengan menggunakan dana yang terbatas. Contoh: Sim City, The Sims, Tamagotchi.

Gambar 2.5 Game simulasi

### 2.2.6 Role Playing

Role Playing. Video game jenis ini sesuai dengan terjemahannya, bermain peran, memiliki penekanan pada tokoh/peran perwakilan pemain di dalam permainan, yang biasanya adalah tokoh utamanya, dimana seiring kita memainkannya, karakter tersebut dapat berubah dan berkembang ke arah yang diinginkan pemain ( biasanya menjadi semakin hebat, semakin kuat, semakin berpengaruh, dll) dalam berbagai parameter yang biasanya ditentukan dengan naiknya **level**, baik dari status kepintaran, kecepatan dan kekuatan karakter, senjata yang semakin sakti, ataupun jumlah teman maupun mahluk peliharaan.Secara kebudayaan, pengembang game Jepang biasanya membuat Role Playing Game (**RPG**) ke arah cerita linear yang diarahkan seolah karakter kita adalah tokoh dalam cerita itu, seperti Final Fantasy, Dragon Quest dan Xenogears. Sedangkan pengembang game **RPG** Eropa, cenderung membuat karakter kita bebas memilih jalan cerita sendiri secara non-linear, seperti Ultima, Never Winter Nights, baldurs gate, Elder Scroll, dan Fallout.


Gambar 2.6 Game role playing

### 2.2.7 Strategi

Strategi. Kebalikan dari video game jenis action yang berjalan cepat dan perlu refleks secepat kilat, video game jenis strategi, layaknya bermain catur, justru lebih memerlukan keahlian berpikir dan memutuskan setiap gerakan secara hati-hati dan terencana. Video game strategi biasanya memberikan pemain atas kendali tidak hanya satu orang tapi minimal sekelompok orang dengan berbagai jenis tipe kemampuan, sampai kendaraan, bahkan hingga pembangunan berbagai bangunan, pabrik dan pusal pelatihan tempur, tergantung dari tema ceritanya. Pemain game strategi melihat dari sudut pandang lebih meluas dan lebih kedepan dengan waktu permainan yang biasanya lebih lama dan santai dibandingkan game action. Unsur-unsur permainannya biasanya berkisar sekitar, prioritas pembangunan, peletakan pasukan, mencari dan memanfaatkan sumberdaya (uang, besi, kayu,minyak,dll), hingga ke pembelian dan peng-upgrade-an pasukan atau teknologi. Game jenis ini terbagi atas:

a. Real time Strategy, game berjalan dalam waktu sebenarnya dan serentak antara semua pihak dan pemain harus memutuskan setiap langkah yang diambil saat itu juga berbarengan mungkin saat itu pihak lawan juga sedang mengeksekusi strateginya. Contoh: Starcraft, Warcraft , dan Command and Conquer.

b. Turn based Strategy , game yang berjalan secara bergiliran, saat kita mengambil keputusan dan menggerakan pasukan, saat itu pihak lawan menunggu, begitu pula sebaliknya, layaknya catur.

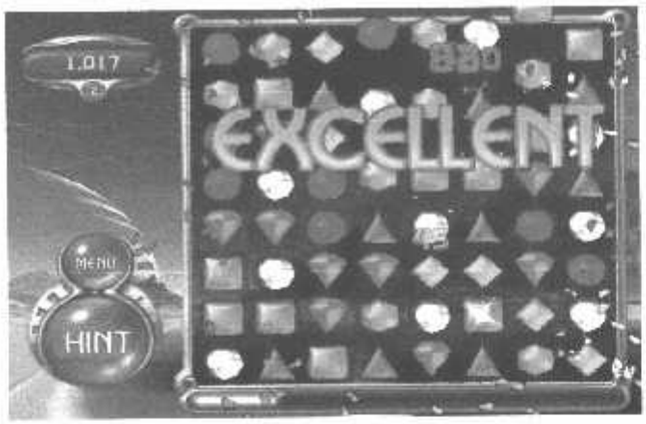
contoh: Front Mission, Super robot wars, Final Fantasy tactics, Heroes of might and magic, Master of Orion.

Sebenarnya ada yang memilah lagi menjadi jenis tactical dan strategi, namun penulis cenderung untuk menggabungkannya karena perbedaannya hanya ada di masalah skala dan ke-kompleks-an dalam manajemen sumber daya-nya saja.

Gambar 2.7 Game strategi

### 2.2.8 Puzzle

Puzzle. Video game jenis ini sesuai namanya berintikan mengenai pemecahan teka-teki, baik itu menyusun balok, menyamakan warna bola, memecahkan perhitungan matematika, melewati labirin, sampai mendorong-dorong kota masuk ke tempat yang seharusnya, itu semua termasuk dalam jenis ini. Sering pula permainan jenis ini adalah juga unsur permainan dalam video game petualangan maupun game edukasi. Tetris, Minesweeper, Bejeweled, Sokoban dan Bomberman.




Gambar 2.8 Game puzzle

### 2.2.9 Simulasi Kendaraan

Simulasi kendaraan. Video Game jenis ini memberikan pengalaman atau interaktifitas sedekat mungkin dengan kendaraan yang aslinya, muskipun terkadang kendaraan tersebut masih eksperimen atau bahkan fiktif, tapi ada penekanan khusus pada detil dan pengalaman realistik menggunakan kendaraan tersebut. Terbagi atas beberapa jenis:

a. Perang. Video game simulasi kendaraan yang sempat tenar di tahun 90-an ini mengajak pemain untuk menaiki kendaraan dan berperang melawan kendaraan lainnya. Dan kebanyakan diantaranya memiliki judul sama dengan nama kendaraannya. Contoh : Apache 64, Comanche, Abrams, YF-23, F-16 fighting eagle.Tetapi game kehidupan bajak laut seperti 'Pirates!' pun dapat dikategorikan disini.

b. Balapan. Dari namanya sudah jelas, siapa sampai duluan di garis finish dialah pemenangnya! Terkadang malah pemain dapat memilih kendaraan, mendandani, upgrade mesin bahkan mengecatnya. Contoh: Top Gear, Test Drive, Sega Rally Championship, Daytona, Grand Turismo, Need For Speed, Mario Cart, ManXTT.

c. Luar Angkasa. Walau masih dapat dikategorikan simulasi kendaraan perang, tetapi segala unsur fiksi ilmiah dan banyaknya judul yang beredar membuat subgenre ini pantas dikategorikan diluar simulasi kendaraan perang. Jenis ini memungkinkan pemain untuk menjelajah luar angkasa, berperang dengan

mahluk alien, mendarat di planet antah berantah atau sekedar ingin merasakan bagaimana menjadi kapten di film fiksi ilmiah kesayangan kamu. Contoh: Wing Commander, Freelancer , Star Wars X-Wing, Star Wars Tie Fighter, dll.

d. *Mecha.* Pendapat bahwa hampir tidak ada orang yang terekspos oleh film robot jepang saat kecilnya tidak memimpikan ingin mengendalikan robot, memang sulit dibantah. Dipopulerkan oleh serial Mechwarrior oleh Activision, subgenre Simulasi Mecha ini memungkinkan pemainnya untuk mengendalikan robot dan menggunakannya untuk menghancurkan gedung, helikopter dan tentu saja robot lainnya. Contoh: Mechwarrior, Gundam Last war Chronicles, dan Armored Core.

Gambar 2.9 Game simulasi kendaraan

### 2.2.10 Olah Raga

Olahraga. Singkat padat jelas, bermain sport di PC atau konsol anda. Biasanya permainannya diusahakan serealistik mungkin walau kadang ada yang menambah unsur fiksi seperti NBA JAM. Contohnya pun jelas, Seri Winning Eleven, seri NBA, seri FIFA, John Madden NFL, Lakers vs Celtics, Tony hawk pro skater, dll.

Gambar 2.10 Game olah raga

## 2.3 Multimedia

Pengertian multimedia ditinjau dari struktur bahasa terdiri atas dua kata, yaitu "multi" yang berarti banyak atau beragam, dan "media" yang berarti perantara atau alat, sehingga jika digabungkan multimedia dapat diartikan sebagai banyak perantara atau beragam perantara. Sedangkan pengertian multimedia ditinjau dari teknologi komputer yaitu penggunaan komputer untuk menyajikan dan menggabungkan teks, suara, gambar, animasi dan video dengan alat bantu (tool) dan koneksi (link) sehingga pengguna dapat ber-(navigasi), berinteraksi, berkarya dan berkomunikasi.

Beberapa definisi multimedia berdasarkan beberapa sumber yaitu :

- Multimedia adalah kombinasi dari komputer dan video (Rosch, 1996)
- Multimedia merupakan kombinasi tiga elemen, yaitu suara, gambar, dan teks (McCormick, 1996)
- Multimedia adalah kombinasi dari paling sedikit dua media input atau output dari data. Media ini dapat audio, animasi, video, teks, grafik, dan gambar (Turban dkk, 2000)
- Multimedia merupakan alat yang dapat menciptakan presentasi yang dinamis dan interaktif yang mengkombinasikan teks, grafik, audio, dan gambar (Robin dan Linda, 2001)
- Multimedia adalah pemanfaatan komputer untuk membuat dan menggabungkan teks, grafik, audio, gambar bergerak dengan menggabungkan

*link* dan *tool* yang memungkinkan pemakai melakukan navigasi, berinteraksi, berkreasi dan berkomunikasi (Hofstetter, 2001)

### 2.3.1 Unsur-unsur Multimedia

Unsur-unsur dalam multimedia adalah :

1. Audio : merupakan suara, musik ataupun bunyi khusus.
2. Video : merupakan gabungan dari beberapa gambar dengan gerakan yang tersambung sehingga gambar terlihat bergerak.
3. Grafik : merupakan gambar suatu objek baik dua dimensi ataupun tiga dimensi yang tidak bergerak (diam).
4. Teks : merupakan huruf, angka, dan simbol-simbol khusus.
5. *Image* : merupakan gambar yang berwarna ataupun hitam putih.

### 2.3.2 Jenis-jenis Multimedia.

Dalam aplikasi multimedia terdapat beberapa jenis yaitu :

1. Internet

   Merupakan suatu media dalam menawarkan suatu produk atau barang secara *on-line*. Pendekatan dengan media ini sangat efisien dan efektif untuk menawarkan barang dan jasa kepada pelanggan sehingga pelanggan dapat dengan mudah memilih produk yang mereka inginkan.

2. Presentasi

   Merupakan suatu media dalam memperkenalkan atau menerangkan suatu produk, laporan, dan lain-lain. Dengan pemakaian unsur multimedia seperti suara dan gerak animasi akan mempermudah pengkomunikasian pesan yang akan disampaikan dan presentasi akan lebih menarik.

3. Film Efek dan Animasi Video

   Pada pembuatan film atau video sekarang ini, banyak menggunakan efek-efek dan animasi-animasi untuk membuat suatu gambar atau adegan yang tidak pernah ada agar lebih menarik.

## 2.4 JAVA

Java menurut definisi dari Sun adalah mana untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer stand alone ataupun pada lingkungan jaringan. Java 2 adalah generasi kedua dari java platform. Kata berdiri di atas sebuah mesin interpreter yang diberi mana Java Virtual Machine (JVM). JVM inilah yang akan membaca bytecode dalam file.class dari suatu program sebagai representasi langsung dari program yang berisi bahasa mesin. Oleh karena itu, bahasa java disebut sebagai bahasa pemrograman yang portable karena dapat dijalankan pada berbagai sistem operasi, asalkan pada sistem operasi tersebut terdapat JVM.

Agar sebuah program Java dapat dijalankan, maka file dengan ekstensi .java harus dikompilasi menjadi file bytecode. Untuk menjalankan bytecode tersebut dibutuhkan JRE (Java Runtime Environment) yang memungkinkan pemakai untuk menjalankan program Java, hanya menjalankan, tidak untuk membuat kode baru lagi. JRE berisi JVM dan library Java yang digunakan.

Java memiliki beberapa versi library atau teknologi yang disebut juga sebagai edisi dari bahasa pemrograman Java. Tiga edisi utama dari library tersebut adalah Micro, Standard, dan Enterprise.

J2ME (Java2 Micro Edition) merupakan edisi library yang dirancang untuk digunakan pada device tertententu seperti pagers dan mobile phone. J2SE (Java2 Standard Edition) merupakan edisi library yang dirancang untuk membuat aplikasi desktop atau applet pada web browser. J2EE (Java2 Enterprise Edition) merupakan edisi librari Java yang dirancang untuk membuat sebuah aplikasi enterprise yang memerlukan antarmuka dengan sumber data (data source) atau dapat pula dikatakan bahwa J2EE adalah kelompok yang lebih besar dengan J2SE di dalamnya.

## 2.5 ECLIPSE

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (*platformindependent*). Berikut ini adalah sifat dari Eclipse:

1. *Multi-platform*: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. *Mulit-language*: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lain seperti C/C++, Coboi, Python, Perl, PHP, dan lain sebagainya.
3. *Multi-role*: Selain sebagai IDE untuk pengembangan aplikasi. Eclipse pun bias digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak seperti dokumentasi, pengujian perangkat lunak, pengembangan web, dan lain sebagainya.

Pada saat ini, Eclipse merupakan salah satu IDE favorit karena gratis dan *open source*. *Open source* berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan membuat komponen yang disebut *plugin*.

**1. Sejarah**

Eclipse awalnya dikembangkan oleh IBM untuk menggantikan perangkat lunak pengembangan IBM Visual Age for Java 4.0. Produk Eclipse ini diluncurkan oleh IBM pada tanggal 5 November 2001. IBM menginvestasikan US$ 40 juta untuk pengembangannya. Sejak 5 November 2001, konsorsium Eclipse Foundation

**2. Arsitektur**

Sejak versi 3.0, Eclipse pada dasarnya merupakan sebuah *kernel*. Apa yang dapat digunakan di dalam Eclipse sebenarnya adalah fungsi dari *plug-in* yang sudah dipasang (diinstal). Ini merupakan basis dari Eclipse yang dinamakan *Rich Client Platform* (RCP). Berikut ini adalah komponen yang membentuk RCP:

- *Core platform*
- OSGi
- SWT (*Standard Widget Toolkit*)
- JFace

- *Eclipse Workbench*

Secara standar Eclipse selalu dilengkapi dengan JDT (*Java Development Tools*), *plug-in* yang membuat Eclipse kompatibel untuk mengembangkan program Java, dan PDE (*Plug-in Development Environment*) untuk mengembangkan *plug-in* baru. Eclipse beserta *plug-in*-nya diimplementasikan dalam bahasa pemrograman Java. Konsep Eclipse adalah IDE adalah

1. terbuka (*open*)
2. mudah diperluas (*extensible*) untuk apa saja
3. tidak untuk sesuatu yang spesifik.

Eclipse tidak saja untuk mengembangkan program Java, tetapi juga untuk berbagai macam keperluan. Perluasan apapun cukup dengan menginstal *plug-in* yang dibutuhkan. Apabila ingin mengembangkan program C/C++ maka telah terdapat *plug-in* CDT (*C/C++ Development Tools*) yang dapat dipasang di Eclipse untuk Eclipse menjadi perangkat untuk pengembangan C/C++.

**3. Histori (versi-versi) Eclipse**

Sejak tahun 2006, Eclipse Foundation mengkoordinasikan peluncuran Eclipse secara rutin dan simultan yang dikenal dengan nama *Simultaneous Release*. Setiap versi peluncuran terdiri dari Eclipse Platform dan juga sejumlah proyek yang terlibat dalam proyek Eclipse.

Tujuan sistem ini adalah untuk menyediakan distribusi Eclipse dengan fitur-fitur dan versi yang terstandarisasi. Hal ini juga dimaksudkan untuk mempermudah *deployment* dan *maintenance* untuk sistem enterprise, serta untuk kenyamanan. Peluncuran simultan dijadwalkan pada bulan Juni setiap tahunnya. Kode Peluncuran Tanggal Peluncuran Platform Nama Proyek

| Kode Peluncuran | Tanggal Peluncuran | Platform | Nama Proyek |
|---|---|---|---|
| Eclipse 3.0 | 28 Juni 2004 | 3.0 | |
| Eclipse 3.1 | 28 Juni 2005 | 3.1 | |

| | | | |
|---|---|---|---|
| Callisto | 30 Juni 2006 | 3.2 | Callisto projects |
| Europa | 29 Juni 2007 | 3.3 | Europa projects |
| Ganymede | 25 Juni 2008 | 3.4 | Ganymede projects |
| Galileo | 24 Juni 2009 | 3.5 | Galileo projects |

Gambar 2.11 Kode peuncuran,platform, dan nama proyek

Gambar 2.12. Tampilan Lembar Kerja Eclipce

## 2.6 ANDROID

Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan platform terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Pada saat perilisan perdana Android, 5 November 2007, Android bersama Open Handset Alliance menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android di

bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler.

Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau Google Mail Services (GMS) dan kedua adalah yang benar–benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai Open Handset Distribution (OHD).

Pada Juli 2000, Google bekerjasama dengan Android Inc., perusahaan yang berada di Palo Alto, California Amerika Serikat. Para pendiri Android Inc. bekerja pada Google, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Saat itu banyak yang menganggap fungsi Android Inc. hanyalah sebagai perangkat lunak pada telepon seluler. Sejak saat itu muncul rumor bahwa Google hendak memasuki pasar telepon seluler. Di perusahaan Google, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yang didukung oleh kernel Linux. Hal ini menunjukkan indikasi bahwa Google sedang bersiap menghadapi persaingan dalam pasar telepon seluler. versi android terbaru yaitu versi 3.0. Android juga sudah bergabung dengan beberapa smart mobile seperti Nokia, Sony Ericsson, dan lainnya.

Sekitar September 2007 sebuah studi melaporkan bahwa Google mengajukan hak paten aplikasi telepon seluler (akhirnya Google mengenalkan **Nexus One**, salah satu jenis telepon pintar GSM yang menggunakan Android pada sistem operasinya. Telepon seluler ini diproduksi oleh HTC Corporation dan tersedia di pasaran pada 5 Januari 2010).

Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android ARM Holdings, Atheros Communications, diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc. Seiring pembentukan Open Handset Alliance, OHA mengumumkan produk perdana mereka, Android, perangkat bergerak (*mobile*) yang merupakan **modifikasi kernel Linux 2.6**. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru.

Telepon pertama yang memakai sistem operasi Android adalah **HTC Dream**, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009

diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android.

### 2.6.1 Kelebihan Android

1. Multitasking –Ponsel Android bisa menjalankan berbagai macam aplikasi, itu artinya Anda bisa browsing, Facebookan sambil dengerin lagu.
2. Kemudahan dalam Notifikasi – Setiap ada SMS, Email, atau bahkan artikel terbaru dari RSS Reader, akan selalu ada notifikasi di Home Screen Ponsel Android, tidak ketinggalan pula Lampu LED Indikator yang berkedip-kedip, sehingga Anda tidak akan terlewatkan satu SMS, Email ataupun Misscall sekalipun.
3. Akses Mudah terhadap Ribuan Aplikasi Android lewat Google Android App Market – Kalau Anda hobi install aplikasi ataupun games, lewat Google Android App Market, Anda bisa mendownload berbagai aplikasi dengan gratis. Ada banyak ribuan aplikasi dan games yang bisa Anda download di ponsel Android.
4. Pilihan Ponsel yang beranekaragam – Bicara ponsel Android,tentu akan teras abeda dibandingkan denagn iOS.jika iOs hanya terbatas pada iPhone dariapple, tetapi ponsel Android tersediadari berbagai produsen, seperti Sony Ericsson, Motorola, HTC, bahkan sampai Samsung. Dan pastinya dari setiap pabrikan ponsel pun menghadirkan ponsel Android dengan gaya masing-masing, seperti Motorola dengan Motoblur-nya, Sony Ericsson dengan TimeScape-nya. Jadi Anda bisa leluasa memilih ponsel Android sesuai dengan 'merk' favorite.
5. Bisa menginstal ROM yang dimodifikasi – jika Anda tidak puas dengan tampilan standar Android, jangan khawatir ada banyak Costum ROM yang bisa Anda pakai di ponsel Android.
6. Widget –dengan adanya Widget di homescreen, Anda bisa dengan mudah mengakses berbagai setting dengan cepat dan mudah.

7. Google Maniak – Jika Anda pengguna setia layanan Google mulai dari Gmail sampai Google Reader, ponsel Android telah terintegrasi dengan layanan Google, schingga Anda bisa dengan cepat mengecek email dari gMail.

### 2.6.2 Kekurangan Android

Dalam segala hal pasti kita tidak akan terlepas dari hal yang berkebalikan. Begitu juga dengan ponsel android ini, jika ada kelebihan pasti juga ada kekurangannya.

1. Koneksi Internet yang terus menerus –kebanyakan ponsel Android memerlukan koneksi internet yang simultan alias terus menerus aktif. Artinya Anda harus siap berlangganan paket GPRS yang sesuai dengan kebutuhan.
2. Iklan – Aplikasi di Ponsel Android memang bisa didapatkan dengan mudah dan gratis, namun Anda harus terima konsekuensinya dari setiap Aplikasi tersebut, akan selalu ada Iklan yang terpampang, entah itu bagian atas atau bawah aplikasi.

# BAB III

# PERANCANGAN DAN ANALISA SISTEM

## 3.1 Pembahasan

Proses perancangan permainan game shooting *Duck Hunter* melalui beberapa tahapan pross sebagai berikut :

1. Perancangan gambar bebek

   Bebek dalam permainan game ini ada 3 macam dimana perbedaan tiap bebek ada pada pertahanan bebek saat di tembak. Jenis bebek pertama akan jatuh jika di tembak sekali, bebek jenis dua akan jatuh di tembak ketika di tembak dua kali, dan jenis bebek ketiga akan jatuh di tembak ketiak bebek ditembak tiga kali.

2. Perancangan gambar senjata

   Senjata yang digunakan dalam game ini terdiri dari :

   1. Small Gunsight (memiliki 3 peluru, waktu yang diperlukan untuk mereload amunisi selama 750 ms).
   2. Small Gunsight (memiliki 5 peluru, waktu yang diperlukan untuk mereload amunisi selama 500 ms, membutuhkan 5000 point untuk mencapai sejanta ini)
   3. Medium Gunsight (memiliki 7 peluru, waktu yang diperlukan untuk mereload amunisi selama 500 ms, membutuhkan 20.000 point untuk mencapai sejanta ini)
   4. Medium Gunsight (memiliki 7 peluru, waktu yang diperlukan untuk mereload amunisi selama 250 ms, membutuhkan 60.000 point untuk mencapai sejanta ini)
   5. Large Gunsight (memiliki 10 peluru, waktu yang diperlukan untuk mereload amunisi instan, membutuhkan 120.000 point untuk mencapai sejanta ini)
   6. Large Gunsight (memiliki 5 peluru, waktu yang diperlukan untuk mereload amunisi instan, membutuhkan 5000 point untuk mencapai sejanta ini)

7. Large Gunsight (memiliki 5 peluru, waktu yang diperlukan untuk mereload amunisi instan, membutuhkan 5000 point untuk mencapai sejanta ini)
8. Large Gunsight (memiliki 5 peluru, waktu yang diperlukan untuk mereload amunisi instan, membutuhkan 5000 point untuk mencapai sejanta ini)
9. Large Gunsight (memiliki 5 peluru, waktu yang diperlukan untuk mereload amunisi instan, membutuhkan 5000 point untuk mencapai sejanta ini)
10. Large Gunsight (memiliki 5 peluru, waktu yang diperlukan untuk mereload amunisi instan, membutuhkan 5000 point untuk mencapai sejanta ini)

Seluruh senjata tersebut memiliki perbedaan jeda waktu pada saat reload peluru/amunisi, dimana semakin bagus senjata yang digunakan maka jeda waktu yang digunakan saat reload peluru/amunisi juga semakin cepat.

3. Perancangan animasi dan suara

Gambar senjata, bebek pada saat jatuh terkena tembakan, gambar background, dan suara telah di dapatkan dari aplikasi game sebelumnya kemudian dimasukan dan disimpan kembali pada software eclipse pada aplikasi game yang penulisan rancang.

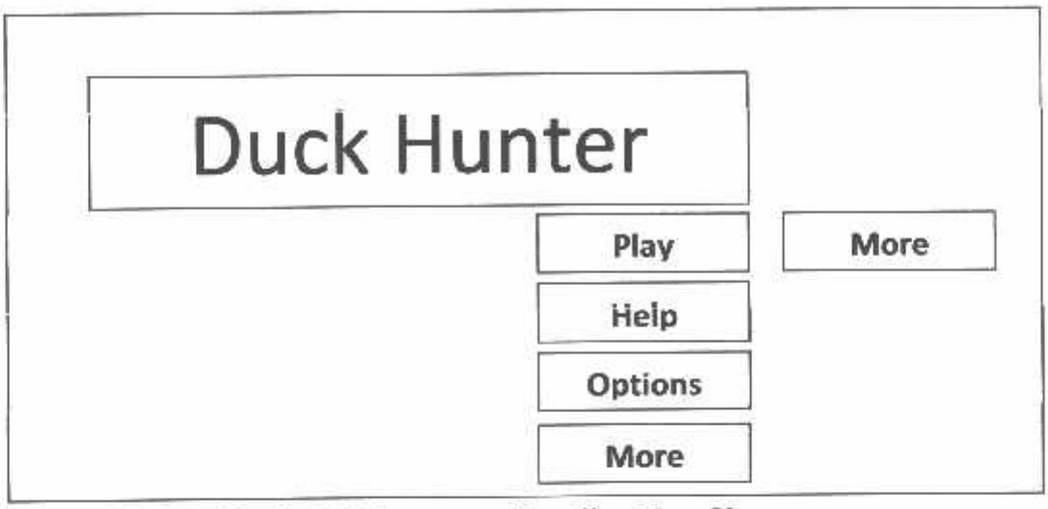
## 3.2 Perancangan

Perangkat lunak permainan Duck Hunter pada handphone dirancang dengan menggunakan bahasa pemrograman *android.*

Perangkat lunak permainan ini beberapa tampilan yaitu :

1. Tampilan Menu Utama.

Halaman ini dirancangdengan menu yang dapat mengaksessmua tampilanlayar program yaitu menu help, menu options, menu score, dan menu level. Tampilan menu utama ini dapat dilihat pada gambar 3.1.



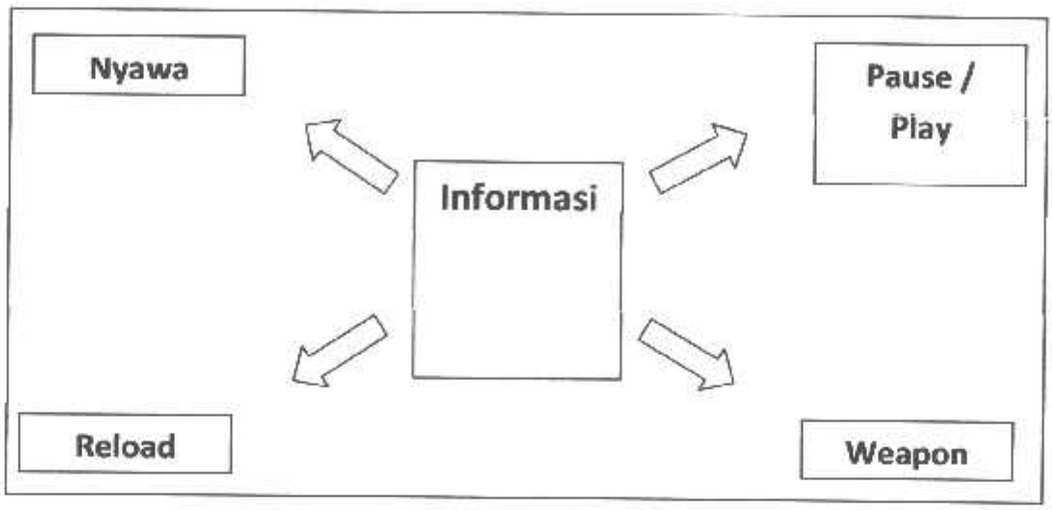

Gambar 3.1 Rancangan Tampilan Menu Utama.

Keterangan Gambar 3.1 :

1. Play : Memulai permainan.
2. Help : bantuan cara bermain.
3. Optioans : mengubah set program.
4. Score : melihat nilai 10 tertinggi yang telah diperoleh dari permainan sebelumnya.
5. About : untuk menuju ke website market android.

2. Tampilan Help.

Halaman ini berisi tetang cara bermain *game*, di sini dijelaskan posisi nyama, sisa amunisi, posisi tombol reload, gambar tembak yang digunakan, dan keadaan pause atau play. Tampilan help ini dapat dilihat pada gambar 3.2.



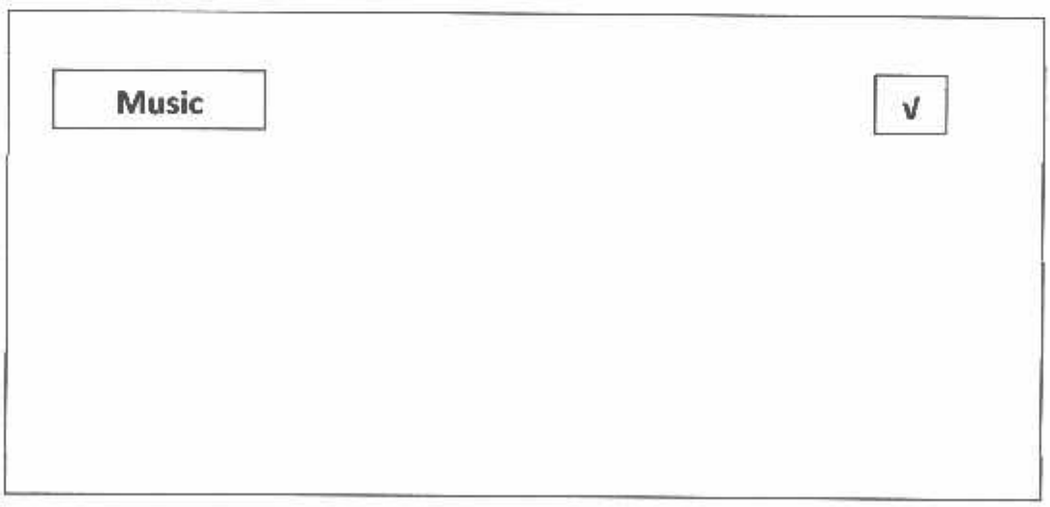

Gambar 3.2 Rancangan Tampilan Help.

Keterangan Gambar 3.2 :

1. Nyawa : Informasi jumlah nyawa pemain.
2. Pause/play : Informasi keadaan permaian berhenti sementara atau berjalan.
3. Reload : Posisi tombol reload.
4. Weapon : Gambar senjata yang digunakan.

3. Tampilan Options.

Halaman ini berisi setting suara apakah suara di hidupkan atau dimatikan di dalam *game*. Tampilan Opstions ini dapat dilihat pada gambar 3.3.



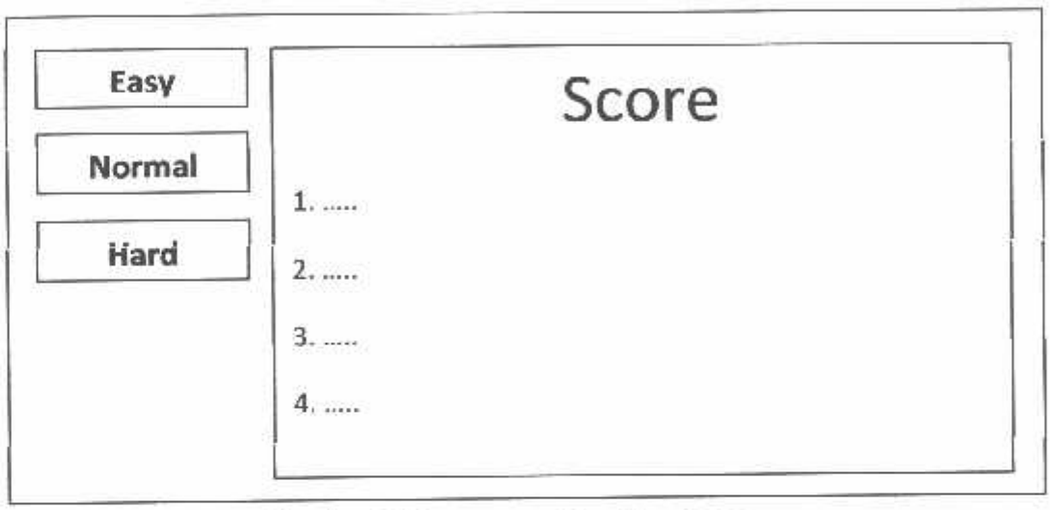
Gambar 3.3 Rancangan Tampilan Options.

Keterangan Gambar 3.3 :

1. Music : mengatur music dalam permainan.

4. Tampilan Score

Halaman ini berisi nilai score 10 besar terbaik yang telah diperoleh dari permainan sebelumnya. Tampilan Score ini dapat dilihat pada gambar 3.4.



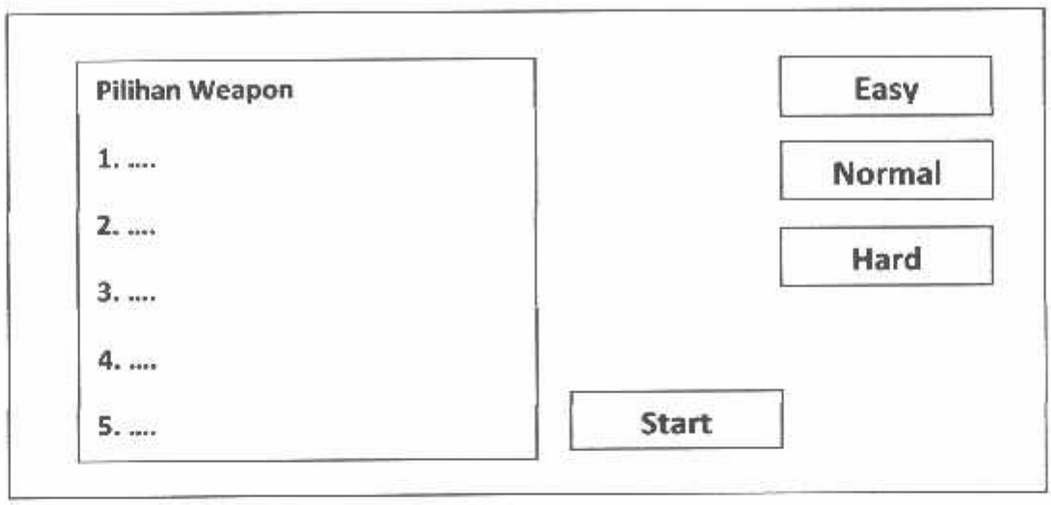

Gambar 3.4 Rancangan Tampilan Score.

Keterangan Gambar 3.4 :

1. Easy : melihat 10 nilai terbaik pada level easy.
2. Normal : melihat 10 nilai terbaik pada level normal.
3. Hard : melihat 10 nilai terbaik pada level hard.
4. Score : Nilai 10 terbaik yang telah diperoleh dari permainan sebelumnya.

5. Tampilan level.

Halaman ini berisi jenis-jenis senjata dan level. Tampilan level ini dapat dilihat pada gambar 3.5.



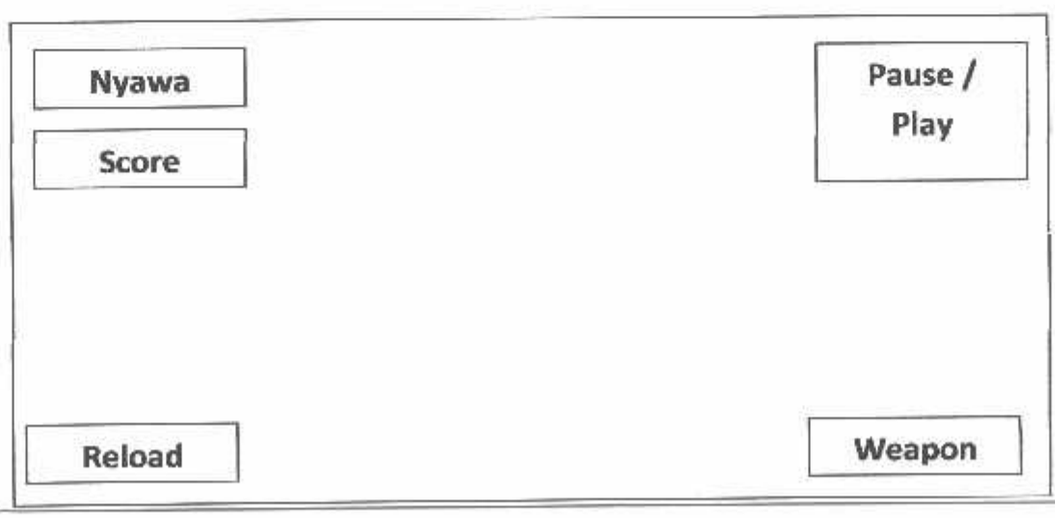

Gambar 3.5 Rancangan Tampilan Score.

Keterangan Gambar 3.5 :

1. Pilihan Weapon : Jenis senjata yang digunakan.
2. Easy : Tingkat kesulitan permanan easy.
3. Normal : Tingkat kesulitan permainan Normal.
4. Hard : Tingkat kesulitan permaian hard.
5. Start : Memulai permainan.

6. Tampilan permainan

Halaman ini berisi permaianan berlasung, pemain dapat melihat nilai yang diperoleh. Tampilan level ini dapat dilihat pada gambar 3.6.



Gambar 3.6 Rancangan Tampilan Permainan.

Keterangan Gambar 3.6 :

1. Nyawa : Informasi jumlah nyawa pemain.
2. Pausc/play : Informasi keadaan permaian berhenti sementara atau berjalan.
3. Reload : tombol reload amunisi.
4. Weapon : Gambar senjata yang digunakan.
5. Score : Nilai yang diperoleh.

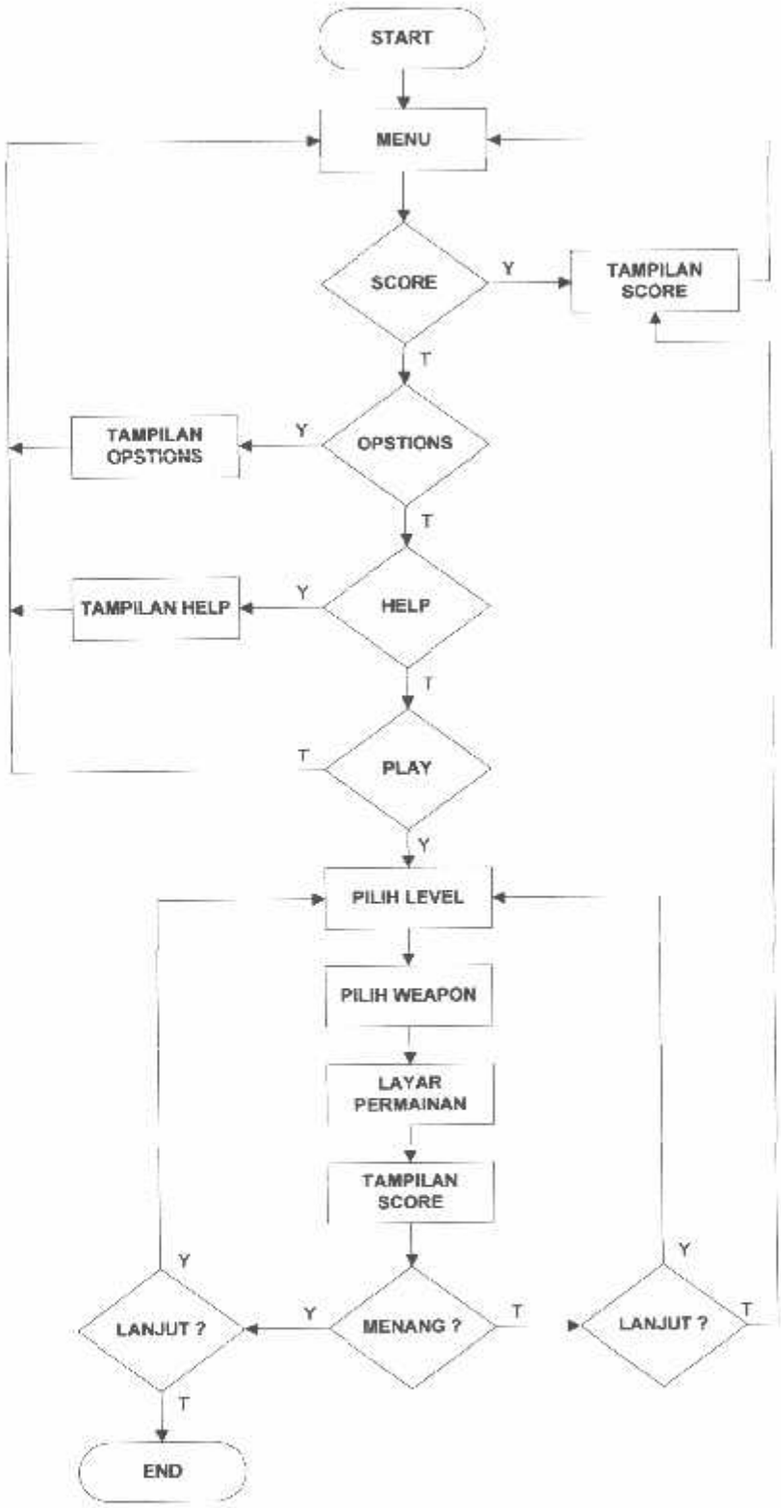
## 3.3 Konsep Game

Langkah awal yang dilakukan dalam pembuatan game adalah konsep atau *storyboard* yang dirumuskan dalam bentuk table. *Storyboard* yang disusun merupakan gambaran umum aplikasi game yang akan dihasilkan melalui tahapan perancangan yang meliupti judul, jenis game, sistem kendali yang digunakan, penentuan target, penentuan background dan gambar sistem permainan. Tabel konsep atau Storyboard dapat di lihat pada Tabel 1.

| | |
|---|---|
| **Judul Game** | Duck hunt |
| **Jenis Game** | Arkade |
| **Jenis Rating** | Ditujukan untuk usia di atas 6 tahun |
| **Sistem Kendali** | Aplikasi game ini menggunakan layar *touchscreen* maka cukup menyentuh layar |
| **Target** | Bebek yang keluar dalam waktu tertentu |
| **Backgroud** | *Static view* dengan ukuran stage 240 x 320 |
| **Sistem Permainan** | - Game ini mempunyai 10 level senjata dimana setiap level senjata tingkat kesulitan berbeda-beda<br>- Pemain harus menembak jatuh seekor bebek yang |

## 3.6 Flowchart

Gambar dibawah ini adalah alur permainan game duck hunter :




*Gambar 3.8 Flowchart Game Duck Hunter.*

# BAB IV
# IMPLEMENTASI DAN PENGUJIAN SISTEM

## 4.1 Implementasi Sistem

Tahap implementasi pengembangan perangkat lunak merupakan proses pengubahan spesifikasi sistem menjadi sistem yang dapat dijalankan. Tahap ini merupakan lanjutan dari proses perancangan, yaitu proses pemrograman perangkat lunak sesuai dengan spesifikasi dan desain sistem.

Pembuatan aplikasi game duck hunter ini menggunakan Java, Android SDK, ADT/Plugins Eclipse, dan Eclipse. Android adalah aplikasi yang dikembangkan dengan berbasis java, sehingga sebelum melakukan *coding* aplikasi berbasis android, komputer/laptop harus sudah terinstal program java.

ADT *(software development kit)* ini diperlukan sebagai alat bantu dan API dalam mengembangkan aplikasi pada platform android menggunakan bahasa pemograman java. Eclipse berfungsi sebagai IDE (*Integrated Development Environment*) menggunakan bahasa pemograman java kemudian dikompilasi bersama data file resource yang dibutuhkan oleh aplikasi, dimana prosesnya di-*package* oleh tool yang dinamakan "apt tools" ke dalam paket android sehingga menghasilkan file dengan ekstensi apk.

### 4.1.1 Kebutuhan Hardware

| Perlengkapan | Spesifikasi | Keterangan |
|---|---|---|
| Handphone | Processor | single core 832Mhz |
| | Memori | 2Gb |
| | Resolusi | 240 x 320 pixel |

Tabel 4.1 Tabel Spesifikasi Perangkat Keras.

Perangkat keras (*hardware*) adalah handphone yang sudah berbasis android.Rincian Perangkat keras (*Hardware*) pendukung seperti dalam Tabel 4.1

### 4.1.2 Kebutuhan Software

Spesifikasi yang digunakan untuk membuat aplikasi ini *minimum requirement* untuk menjalankan aplikasi adalah android dengan sistem operasi android 2.3.5 (Gingerbread).

## 4.2 Pengujian Sistem

Perangkat lunak ini didesain pada sistem operasi android 2.3.6 (Gingerbread) dengan resolusi monitor 240 x 320 pixel. Oleh karena itu, akan diuji dengan menggunakan beberapa handphone berbeda-beda dengan resolusi monitor yang berbeda – beda juga.

Pengujian pertama dilakukan dengan spesifikasi perangkat keras dan perangkat lunak sebagai berikut :

1) Handphone : Samsung Galaxy Young (GT-S5360)
2) Processor : single core 832Mhz
3) Memory : 2Gb
4) Resolusi : 240 x 320 pixel
5) Sistem Operasi : Gingerbread
6) Versi Android : 2.3.6

### 4.2.1. Pengujian *Menu Utama*

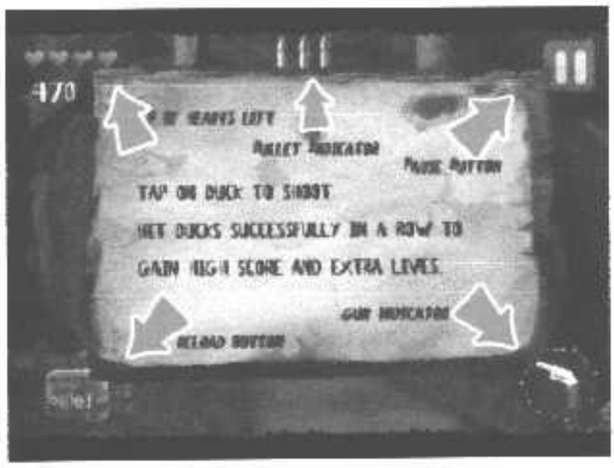
Bagian ini adalah bagian yang pertama kali keluar saat game di nyalakan, setelah game duck hunter di nyalakan seorang pemain dapat melihat tombol *Play, Help, Options, Score,* dan *About*. Tombol *Play* berfungsi untuk menuju ke halaman level, tombol *Help* berfungisi untuk menuju ke halaman *Help* dimana di halaman tersebut akan dijelaskan letak posisi tombol reload, gambar senjata yang dipakai, jumlah darah pemain, sisa amunisi, dan tombol pause, tombol *Options* berfungsi utnuk menuju halaman *Options* dimana di dalam halaman *Options* pemain dapat mensetting suara, tombol Score berfungsi menuju ke halaman *Score* dimana didalam halaman *Score* akan di jelaskan nilai 10 terbaik yang telah diperoleh dari permainan sebelumnya, dan tombol *About* berfungsi untuk menuju ke halaman *About* yang menceritakan tentang aplikasi game ini.

Halaman *menu utama* di buat dengan sangat menarik karena fungsi dari halaman ini adalah untuk dilihat pengunjung secara umum. Tampilan menu utama ini dapat dilihat pada gambar 4.1.


Gambar 4.1 Tampilan Halaman Menu Utama.

### 4.2.2. Pengujian *Help*

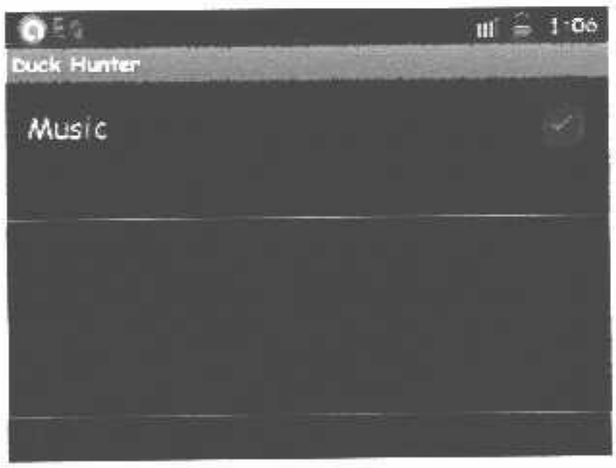
Bagian ini adalah bagian yang memberikan informasi letak posisi tombol-tombol yang digunakan saat memainkan game, daalam halaman ini pemain akan ditunjukan posisi letak posisi tombol reload yang berfungsi untuk mengisi ulang amunisi, gambar senjata yang berfungsi untuk memberikan informasi senjata yang terpakai didalam game, jumlah darah yang berfungsi untuk memberikan informasi sisa darah pemain, gambar amunisi yang berfungsi untuk memberikan informasi sisa amunisi agar pemain bias bersiap-siap untuk mereload apabila amunisi habis, dan tombol pause berfungsi untuk memberikan informasi keadaan game dalam keadaan pause atau sedang berjalan. Tampilan menu utama ini dapat dilihat pada gambar 4.2.




Gambar 4.2 Tampilan Halaman Help.

### 4.2.3. Pengujian *Options*

Bagian ini adalah bagian untuk merubah settinggan didalam aplikasi game duck hunter. Didalam halaman *Options* pemain dapat merubah settingan suara diaktifkan atau di nonaktifkan, apabila suara diaktifkan maka akan muncul suara pada saaat game berjalan begitu juga sebaliknya apabila suara dinonaktifkan maka tidak muncul suara selama game berjalan. Tampilan *Options* ini dapat dilihat pada gambar 4.3.




Gambar 4.3 Tampilan Halaman Options.

### 4.2.4. Pengujian *Score*

Bagian ini adalah bagian yang menunjukan nilai terbaik pada game duck hunter, didalam halaman ini pemain akan ditunjukan 10 nilai terbaik yang telah diperoleh dari permainan sebelumnya menurut tingkat kesulitan (easy, normal, hard). Tampilan *Score* ini dapat dilihat pada gambar 4.4.


Gambar 4.4 Tampilan Halaman Score.

### 4.2.5. Pengujian *Level*

Bagian ini adalah bagian yang berisis jenis-jenis senjata dan tingkat kesulitan (easy, normal, hard), didalam halaman ini pemain dapat memilih senjata yang akan digunakan, setiap senjata mempunyai syarat poin apabila poin pemain belum mencukupi maka senjata tersebut masi terkunci dan pemain juga dapa memilih tingkatan kesulitan (easy, normal, hard) di dalam halaman ini. Didalam halaman ini juga terdapat tombol start yang berfungsi untuk memulai game. Tampilan *Level* ini dapat dilihat pada gambar 4.5.


Gambar 4.5 Tampilan Halaman Level.

### 4.2.6. Pengujian *Permainan*

Bagian ini adalah bagian permaianan yang sedang berlangsung, didalam halaman ini pemain dapat bermain dengan cara menembak bebek. Tampilan *Level* ini dapat dilihat pada gambar 4.6.


Gambar 4.6 Tampilan Halaman Permainan.

## 4.3 Pengujian Aplikasi

| Pengujian | WT19i | GT-S5360 |
|---|---|---|
| Kualitas Grafik | Sangat Baik | Sangat Baik |
| Kualitas Suara | Sangat Baik | Baik |
| Kestabilan Game | Sangat Baik | Sangat Baik |

Tabel 4.2 Tabel Uji Coba Aplikasi

Pada uji coba aplikasi game ini telah dilakukan pada 2 handphone dengan merek dan system operasi yang berbeda yaitu pada handphone sony ericsson WT19i dengan system operasi android 3.0 (honeycomb) dan Samsung GT-S5360 dengan system operasi android 2.3.6 (Gingerbread). Hasil uji coba dari 2 handphone dapat dilihat pada table 4.2.

## 4.4 Pengujian Terhadap User

| No | Nilai | Jumlah User |
|---|---|---|
| 1 | Sangat Baik | 7 |
| 2 | Baik | 2 |
| 3 | Kurang | 1 |

Tabel 4.3 Tabel Nilai Uji Coba Aplikasi Terhadap User

Pada uji coba aplikasi game ini telah dilakukan dengan bantuan 10 (sepuluh) *user*/pemain untuk memperoleh hasil berupa data tentang kelayakan aplikasi ini baik dari segi tampilan maupun isi. Rincian nilai untuk aplikasi ini, ditunjukkan dalam tabel 4.4.

| User | Komentar | | |
|---|---|---|---|
| | Kurang | Baik | Sangat Baik |
| 1 | | 2 | 4 |
| 2 | 1 | 2 | 3 |
| 3 | | 3 | 3 |
| 4 | | 5 | 1 |
| 5 | | 4 | 2 |
| 6 | 1 | 3 | 2 |
| 7 | | 5 | 1 |
| 8 | | 1 | 5 |
| 9 | | 3 | 3 |
| 10 | | 2 | 4 |
| Nilai Total | 2 | 30 | 28 |

Tabel 4.4 Tabel Data Hasil Kuisioner

1. apakah kamu suka bermain game shooting ?

   a.ya

   b.tidak

2. apa yang kamu ketahui tetang game duck hunter ?

   a.menantang

   b.biasa saja

   c.membosankan

3. apakah kamu menyukai fitur baru game duck hunter ?

   a.sangat suka

   b.suka

   c.tidak suka

4. bagaimana menurut kamu tentang kualitas grafik dari game duck hunter ?

   a.sangat bagus

   b.bagus

   c.cukup

   d.kurang bagus

5. apakah kamu mendapatkan tantangan setelah memainkan game duck hunter ?

   a.sangat menantang

   b.cukup menantang

   c.tidak ada tantangan

6. menurut kamu, bagaimana perkembangan game duck hunter yang telah di buat oleh penulis ?

   a.sangat bagus

   b.bagus

   c.cukup

   d.kurang bagus

Setelah mencoba aplikasi *user* diberikan lembar kuisioner untuk diisi, dimana kuisioner tersebut berisikan 6 (enam) pertanyaan seputar komentar tentang kelayakan aplikasi game duck hunter ini. Dari hasil kuisioner diatas diperoleh data yang di tunjukkan dalam tabel 4.5.

# BAB V
# PENUTUP

## 5.1 Kesimpulan

1. Aplikasi game duck hunter ini dapat digunakan pada handphone manapun dengan resolusi yang berbeda-beda yang mendukung sistem operasi Android.
2. Kelebihan dari aplikasi game duck hunter yang dibuat oleh penulis ini adalah dengan adanya tambahan jenis senjata dan penigkatan tingkat kesulitan yang akan membuat aplikasi game duck hunter ini menjadi semakin menantang untuk dimainkan.

## 5.2 Saran

1. Perlu adanya tentang kualitas teknis aplikasi game duck hunter ini masih perlu dikembangkan kearah yang lebih baik lagi seperti penambahan jenis bebek berbeda sehingga akan banyak variasi baru untuk menuju kesempurnaan program.
2. Kualitas program yang dapat di perbaiki diantaranya penggunaan grafik yang lebih bagus dengan resolusi yang besar agar lebih jelas.

# DAFTAR PUSTAKA


1. http://www.edu4java.com/androidgame.html.
2. http://id.wikipedia.org/wiki/Eclipse_(perangkat_lunak).
3. http://en.wikipedia.org/wiki/Duck_Hunt.
4. http://putrikhairanikoto.blogspot.com/2011/05/konsep-android.html.
5. http://nasuma.blogspot.com/2012/02/android-arsitektur-konsep-konsep-kunci.html.
6. Maxikom, Java untuk orang awam.
7. Bamboomedia, Bikin Game HP.
8. Nazruddin Safaat H, Pemograman Aplikasi mobile Smartphone dan Tablet PC Berbasis Android, Informatika.
9. Firdan Ardiansyah, Pengenalan Dasar Android Programming.
10. Mario Zechmer, Beginning Android Games.

# LAPIRAN CODING

```
package game.berburu.bebek.sprite;

import android.content.res.Resources;
import android.graphics.Canvas;

public abstract class AbstractDuck extends AbstractSprite
{
 protected int costMoney;
 protected int duckCount = 0;
 protected int duck_type;
 protected boolean isRight;
 protected boolean isshoot;
 protected int liveTime;
 protected Orbit orbit;
 protected int shootNumber = 0;
 protected long startTime;
 protected long stayTime;
 protected long time;

 public AbstractDuck(Resources paramResources)
 {
  super(paramResources);
 }

 public void addShootNumber()
 {
  this.shootNumber = (1 + this.shootNumber);
 }

 public abstract void calcFrame(long paramLong);

 public abstract void drawFrame(Canvas paramCanvas);

 public int getCostMoney()
 {
  return this.costMoney;
 }

 public int getDuck_type()
 {
  return this.duck_type;
 }
```

```
public abstract void getFirstDrawable();

public int getLiveTime()
{
 return this.liveTime;
}

public Orbit getOrbit()
{
 return this.orbit;
}

public int getShootNumber()
{
 return this.shootNumber;
}

public long getStayTime()
{
 return this.stayTime;
}

public void initDuckCount()
{
 this.duckCount = 0;
}

public boolean isDisPlay()
{
 return this.orbit.isDisply();
}

public boolean isIsshoot()
{
 return this.isshoot;
}

public boolean isRight()
{
 return this.isRight;
}

public void resumePause()
{
 this.startTime = System.currentTimeMillis();
```

```
}

public void setCostMoney(int paramInt)
{
 this.costMoney = paramInt;
}

public void setDuck_type(int paramInt)
{
 this.duck_type = paramInt;
 this.costMoney = (paramInt * 10);
}

public void setIsshoot(boolean paramBoolean)
{
 this.time = 0L;
 this.duckCount = 0;
 this.orbit.setXs(0.0D);
 this.isshoot = paramBoolean;
}

public void setLiveTime(int paramInt)
{
 this.liveTime = paramInt;
}

public void setOrbit(Orbit paramOrbit)
{
 this.orbit = paramOrbit;
}

public void setPausetTime(long paramLong)
{
 this.orbit.setPauseTime(paramLong);
}

public void setRight(boolean paramBoolean)
{
 this.duckCount = 0;
 this.isRight = paramBoolean;
}

public void setShootNumber(int paramInt)
{
 this.shootNumber = paramInt;
}
```

```
 public void setStayTime(long paramLong)
 {
  this.stayTime = paramLong;
 }
}
```

---

```
package game.berburu.bebek.sprite;

import android.content.res.Resources;
import android.graphics.Canvas;
import android.graphics.Rect;

public abstract class AbstractSprite
 implements Cloneable
{
 protected int _centerRectX;
 protected int _centerRectY;
 protected int _centerX;
 protected int _centerY;
 protected int _height = 60;
 protected Resources _resources;
 protected int _speedX;
 protected int _speedY;
 protected int _width = 70;
 protected int _x;
 protected int _y;
 protected boolean displayed;
 protected boolean isDisplay;
 protected boolean isSend;
 protected Rect rect = new Rect(0, 0, 0, 0);
 protected Rect rectShoot = new Rect(0, 0, 0, 0);
 protected Rect rectUnshoot = new Rect(0, 0, 0, 0);

 public AbstractSprite(Resources paramResources)
 {
  this._resources = paramResources;
 }

 public abstract void calcFrame(long paramLong);

 public boolean containUnXY(float paramFloat1, float paramFloat2)
 {
```

```
  return this.rectUnshoot.contains((int)paramFloat1, (int)paramFloat2);
 }

 public boolean containXY(float paramFloat1, float paramFloat2)
 {
  return this.rectShoot.contains((int)paramFloat1, (int)paramFloat2);
 }

 public abstract void drawFrame(Canvas paramCanvas);

 public Rect getRect()
 {
  return this.rect;
 }

 public Rect getRectShoot()
 {
  return this.rectShoot;
 }

 public Rect getRectUnshoot()
 {
  return this.rectUnshoot;
 }

 public int get_centerX()
 {
  return this._centerX;
 }

 public int get_centerY()
 {
  return this._centerY;
 }

 public int get_height()
 {
  return this._height;
 }

 public Resources get_resources()
 {
  return this._resources;
 }

 public int get_width()
```

```
  {
    return this._width;
  }

  public int get_x()
  {
    return this._x;
  }

  public int get_y()
  {
    return this._y;
  }

  public boolean isSend()
  {
    return this.isSend;
  }

  public void setRect()
  {
    setRectVoid();
    setRectRectShoot();
    setUnRectVoid();
  }

  ublic abstract void setRectRectShoot();

  ublic void setRectShoot(Rect paramRect)

    this.rectShoot = paramRect;


  ublic void setRectUnshoot(Rect paramRect)

    this.rectUnshoot = paramRect;


  ublic void setRectVoid()

    this.rect.set(this._x, this._y, this._x + this._width, this._y + this._height);
    this._centerX = this.rect.centerX();
    this._centerY = this.rect.centerY();


  ublic void setSend(boolean paramBoolean)
```

```
 {
  this.isSend = paramBoolean;
 }

 public abstract void setUnRectVoid();

 public void set_centerX(int paramInt)
 {
  this._centerX = paramInt;
 }

public void set_centerY(int paramInt)
{
 this._centerY = paramInt;
}

public void set_height(int paramInt)
{
 this._height = paramInt;
}

ublic void set_resources(Resources paramResources)

this._resources = paramResources;

ublic void set_width(int paramInt)

his._width = paramInt;

ublic void set_x(int paramInt)

his._x = paramInt;

iblic void set_y(int paramInt)

his._y = paramInt;
```

---

```
kage game.berburu.bebek.sprite;

ort game.berburu.bebek.ResourceManager;
```

```
import android.R;
import android.content.res.Resources;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Rect;
mport android.graphics.drawable.BitmapDrawable;


ublic class Background extends AbstractSprite

private static final int[] back_ground;
private int _altitude;
rivate Bitmap _bitmap;
rivate Paint _paint = new Paint();


tatic

int[] arrayOfInt = new int[1];
arrayOfInt[0] = R.id.background;
back_ground = arrayOfInt;


iblic Background(Resources paramResources)

uper(paramResources);
etBitmap(back_ground[0]);


vate void getBitmap(int paramInt)

is._bitmap = ResourceManager.getDrawable(this._resources, paramInt).getBitmap();

lic void calcFrame(long paramLong)


lic void dragCanvasDown(int paramInt)

s._altitude = (paramInt + this._altitude);

lic void drawFrame(Canvas paramCanvas)
```

```
{
 Rect localRect1 = new Rect(0, 0, this._bitmap.getWidth(), this._bitmap.getHeight());
 Rect localRect2 = new Rect(0, 0, 480, 320);
 paramCanvas.drawBitmap(this._bitmap, localRect1, localRect2, this._paint);
}

public void setRectRectShoot()
{
}

public void setUnRectVoid()
{
}
```

---

```
ckage game.berburu.bebek.sprite;

port game.berburu.bebek.R;
port game.berburu.bebek.ResourceManager;
port android.content.res.Resources;
port android.graphics.Bitmap;
port android.graphics.Canvas;
port android.graphics.drawable.BitmapDrawable;


lic class Ball extends AbstractSprite

ivate BallInfo ballInfo;
ivate BitmapDrawable imageDrable = ResourceManager.getDrawable(this._resources,
awable.bullet);

blic Ball(Resources paramResources)

per(paramResources);


vate void getDrawable(int paramInt)

is.imageDrable = ResourceManager.getDrawable(this._resources, paramInt);


lic void calcFrame(long paramLong)

Drawable(R.drawable.bullet);
```

```
public void drawFrame(Canvas paramCanvas)
{
 this._x = ((480 - this.imageDrable.getBitmap().getWidth() * this.ballInfo.getMaxCount()) / 2);
 int i = 0;
 if (i >= this.ballInfo.getCurrentCount());
 for (int j = this.ballInfo.getCurrentCount(); ; j++)
 {
  if (j >= this.ballInfo.getMaxCount())
  {

   getDrawable(R.drawable.bullet);
   this._width = this.imageDrable.getBitmap().getWidth();
   this._height = this.imageDrable.getBitmap().getHeight();
   setRectVoid();
   this.imageDrable.setBounds(this.rect);
   this.imageDrable.draw(paramCanvas);
   this._x = (this.imageDrable.getBitmap().getWidth() + this._x);

   break;
 }
 getDrawable(R.drawable.unbullet);
 this._width = this.imageDrable.getBitmap().getWidth();
 this._height = this.imageDrable.getBitmap().getHeight();
 setRectVoid();
 this.imageDrable.setBounds(this.rect);
 this.imageDrable.draw(paramCanvas);
 this._x = (this.imageDrable.getBitmap().getWidth() + this._x);
}

ublic BallInfo getBallInfo()

eturn this.ballInfo;


ublic void setBallInfo(BallInfo paramBallInfo)

his.ballInfo = paramBallInfo;
his._x = (240 - this.imageDrable.getBitmap().getWidth() * paramBallInfo.getMaxCount() /

his._y = 10;


blic void setRectRectShoot()
```

```
  }

  public void setUnRectVoid()
  {
  }
}
```

---

```
ackage game.berburu.bebek.sprite;

ublic class BallInfo

private int currentCount;
rivate boolean isReloading;
rivate int maxCount;

ublic void deleteBall()

this.currentCount -= 1;

ublic void full()

his.currentCount = this.maxCount;

blic int getCurrentCount()

turn this.currentCount;

blic int getMaxCount()

turn this.maxCount;

lic boolean isEmpty()

(this.currentCount == 0)
 return false;
 else return true;

lic boolean isFull()

this.currentCount == this.maxCount)
```

```
        return true;
    else return false;
}

public boolean isReloading()
{
  return this.isReloading;
}

public void setMaxCount(int paramInt)
{
  this.maxCount = paramInt;
}

ublic void setReloading(boolean paramBoolean)
{
 this.isReloading = paramBoolean;
```

---

```
ckage game.berburu.bebek.sprite;

port game.berburu.bebek.R;
port game.berburu.bebek.SoundManager;
port game.berburu.bebek.utility.AnimationSeries;
port android.app.Activity;
port android.view.View;

lic class BoardFallDown

ivate static final int[] anims;

tic

t[] arrayOfInt = new int[2];
rrayOfInt[0] = R.anim.falldown;
rrayOfInt[1] = R.anim.shake;
nims = arrayOfInt;


blic static void startAnimation(Activity paramActivity,  View paramView)

t[] arrayOfInt = anims;
unnable[] arrayOfRunnable = new Runnable[2];
rayOfRunnable[0] = new Runnable( )
```

```
    {
      public void run()
      {
      // BoardFallDown.playSoundEffect(SoundManager.Type.WoodBoardFallDown);
      }
    };
    new AnimationSeries(paramActivity, paramView, arrayOfInt, arrayOfRunnable).start();
  }
```

---

```
ıckage game.berburu.bebek.sprite;

ıport android.content.res.Resources;
ıport android.graphics.Bitmap;
ıport android.graphics.Canvas;
ıport android.graphics.Paint;
port android.graphics.Rect;

blic class BullEye extends AbstractSprite

rivate Gun.GunSight _gunSight;
·ivate double _length;
ivate float _xs;
ivate float _ys;
otected Rect drawRect = new Rect(0, 0, 0, 0);
ivate Bitmap image;
ivate long nowTime;
ivate long pasueTime = 0L;

blic BullEye(Resources paramResources)

ıper(paramResources);


)lic void calcFrame(long paramLong)

ng l1 = 0L;
(this.nowTime == 0L)

his.nowTime = paramLong;
f (this.pasueTime > 0L)

if (l1 <= 0L)
```

```
        l1 -= this.pasueTime;
      }
    }
    for (this.pasueTime = 0L; ; this.pasueTime = 0L)
    {
     if ((l1 > 0L) && (l1 < 50L)) {
          return;
     }
     long l2 = System.currentTimeMillis();
     l1 = l2 - this.nowTime;
     this.nowTime = l2;
     break;
   }
 }

public void drawFrame(Canvas paramCanvas)

 this._x -= this._width / 2;
 this._y -= this._height / 2;
 setRectVoid();
 paramCanvas.drawBitmap(this.image, this.drawRect, this.rect, new Paint());
 this._x += this._width / 2;
 this._y += this._height / 2;


ublic long getPasuTime()

eturn this.pasueTime;


blic double get_length()

turn this._length;


blic float get_xs()

turn this._xs;


lic float get_ys()

urn this._ys;


ic void initGun()
```

```
{
 this.image = this._gunSight.getDrawable(this._resources);
 this._width = this.image.getWidth();
 this._height = this.image.getHeight();
 this._x = 240;
 this._y = 160;
 this.drawRect.set(0, 0, this._width, this._height);
}

public void setGunSight(Gun.GunSight paramGunSight)
{
 this._gunSight = paramGunSight;
}

ublic void setPasuTime(long paramLong)

this.pasueTime = paramLong;


ublic void setRectRectShoot()


ublic void setUnRectVoid()


iblic void set_length(double paramDouble)

iis._length = paramDouble;


blic void set_xs(float paramFloat)

iis._xs = paramFloat;


olic void set_ys(float paramFloat)

is._ys = paramFloat;
```

---

```
age game.berburu.bebek.sprite;
```

```
ublic class Constants
{
 public static final int BALL_WIDTH = 9;
 public static final float BALL_X_SPEED = 10.2F;
 public static final float BALL_Y_SPEED = 8.2F;
 public static final String FORWARD_TO_GAME = "forward_to_game";
 public static final int FRAME_LAST_MS = 50;
 public static final String GAME_MODE = "game_mode";
 public static final Gun[] GUNS;
 ublic static final String GUN_LEVEL = "gun";
 ublic static final String LEVEL = "level";
 ublic static final String LIFE = "life";
 ublic static final int LIFE_INIT_NUM = 3;
 ublic static final int LIFE_MAX_NUM = 5;
 ublic static final int LOGIC_BEYE_HEIGHT = 40;
 ublic static final int LOGIC_BEYE_WIDTH = 40;
 ublic static final int LOGIC_BULLY_LEGHT = 20;
 ublic static final int LOGIC_DUCK_HEIGHT = 60;
 ublic static final int LOGIC_DUCK_WIDTH = 70;
 iblic static final int LOGIC_GAME_BALLA_HEIGHT = 285;
 iblic static final int LOGIC_GAME_HALF_WIDTH = 240;
 iblic static final int LOGIC_GAME_HEIGHT = 320;
 iblic static final int LOGIC_GAME_WIDTH = 480;
 blic static final int LOGIC_GAME_ZEOR = 0;
 blic static final int LOGIC_READ_TIME = 5;
 blic static final int LOGIC_TIME = 16;
 blic static final int MAXHEIGHT = 80;
 blic static final int MAXLEFT = 80;
 olic static final int MAXLEFT_SPACE = 5;
 olic static final GameMode[] MODE;
 olic static final String MONEY = "momey";
 olic static final int NEXT_LEVEL = 103;
 olic static final int SORCE = 10;
 olic static final int UPDATE_GUN = 102;
 olic static final int UPDATE_LIFE = 101;
 lic static final int UPDATE_SORCE = 100;
 lic static final String WIN = "win";

ic

meMode[] arrayOfGameMode = new GameMode[3];
ayOfGameMode[0] = GameMode.EASY;
ayOfGameMode[1] = GameMode.MIDDLE;
ayOfGameMode[2] = GameMode.HARD;
)DE = arrayOfGameMode;
n[] arrayOfGun = new Gun[5];
```

```
    arrayOfGun[0] = Gun.GUN1;
    arrayOfGun[1] = Gun.GUN2;
    arrayOfGun[2] = Gun.GUN3;
    arrayOfGun[3] = Gun.GUN4;
    arrayOfGun[4] = Gun.GUN5;
    GUNS = arrayOfGun;
}
```

---

```
ackage game.berburu.bebek.sprite;

iport game.berburu.bebek.GameControl;

iport java.util.ArrayList;

blic class DuckFactory

ublic static DuckFactory INSTANCE = new DuckFactory();

ivate int getLevelMax(int paramInt)

nt i;
f (paramInt > 10)
i = 2;
lse
((paramInt <= 10) && (paramInt > 5))

i = 1;
else i =0;
turn i;


lic ArrayList<AbstractSprite> generalDuck(GameControl paramGameControl)

meLevel localGameLevel = paramGameControl.get_gameActivity().get_level();
i = localGameLevel.getLevel() * (1 +
nGameControl.get_gameActivity().getGame_Mode()) + localGameLevel.getMinlevel() / 2;
rayList localArrayList = new ArrayList(i);
j = localGameLevel.getMinlevel();
g l = 0L;
(int k = 0; ; k++)

(k >= i)
eturn localArrayList;
```

```
    AbstractDuck localAbstractDuck = (AbstractDuck)randomDuck(j, paramGameControl);
    localArrayList.add(localAbstractDuck);
    localAbstractDuck.setStayTime(l);
    l = l + 1000 * Tools.randomInt(1, 4) * Tools.randomInt(2) + 1000 * Tools.randomInt(2);
    localAbstractDuck.setLiveTime((int)(0.9D * (16 -
aramGameControl.get_gameActivity().getGame_Mode() - localGameLevel.getMinlevel() / 2 -
etLevelMax(localGameLevel.getLevel())));
  }
}

public AbstractSprite randomDuck(int paramInt, GameControl paramGameControl)
{
 FlyDuck localFlyDuck = new
yDuck(paramGameControl.get_gameActivity().getResources());
 int i;
 if (paramInt == 1)
  i = 1;
 else if (paramInt == 2)
  {
   i = 2;

  }
lse  if (paramInt == 3)
 {
  i = 3;

 }
lse
i = 3;

localFlyDuck.setDuck_type(1 + Tools.randomInt(i));
FlyOrbit localFlyOrbit = new FlyOrbit(localFlyDuck.get_width(),
lFlyDuck.get_height(),
mGameControl.get_gameActivity().get_mode().getFly_minspeed(),
mGameControl.get_gameActivity().get_mode().getFly_maxspeed(),
mGameControl.get_gameActivity().get_level(),
mGameControl.get_gameActivity().get_mode());
ocalFlyDuck.setOrbit(localFlyOrbit);
ocalFlyDuck.getFirstDrawable();
ocalFlyOrbit.setDisply(true);
ocalFlyOrbit.init();
eturn localFlyDuck;
```

```
)ackage game.berburu.bebek.sprite;

mport game.berburu.bebek.R;
mport game.berburu.bebek.ResourceManager;
nport android.content.res.Resources;
nport android.graphics.Bitmap;
nport android.graphics.Canvas;
nport android.graphics.ColorMatrixColorFilter;
1port android.graphics.Paint;
1port android.graphics.Rect;
1port android.graphics.drawable.BitmapDrawable;


blic class FlyDuck extends AbstractDuck

rivate static final int[] duck1_dead;
rivate static final int[] duck2_dead;
rivate static final int[] duck3_dead;
:ivate static final int[] duck_fly;
ivate static final int[] duck_fly_double_1;
ivate static final int[] duck_fly_double_2;
ivate static final int[] duck_fly_three_1;
ivate static final int[] duck_fly_three_2;
ivate static final int[] duck_fly_three_2_block;
ivate static final int[] duck_fly_three_3;
vate static final int[] duck_fly_three_block;
vate static final int[][] ducks_dead;
vate BitmapDrawable _currentDrawable;
vate Rect rec = new Rect();

ic

![] arrayOfInt1 = new int[4];
rayOfInt1[0] = R.drawable.bird_1;
'ayOfInt1[1] = R.drawable.bird_2;
'ayOfInt1[2] = R.drawable.bird_3;
ayOfInt1[3] = R.drawable.bird_4;
ck_fly = arrayOfInt1;
[] arrayOfInt2 = new int[4];
ayOfInt2[0] = R.drawable.bird_elite_1;
ayOfInt2[1] = R.drawable.bird_elite_2;
ayOfInt2[2] = R.drawable.bird_elite_3;
ayOfInt2[3] = R.drawable.bird_elite_4;
k_fly_double_1 = arrayOfInt2;
```

```
int[] arrayOfInt3 = new int[4];
arrayOfInt3[0] = R.drawable.bird_elite_injured_1;
arrayOfInt3[1] = R.drawable.bird_elite_injured_2;
arrayOfInt3[2] = R.drawable.bird_elite_injured_3;
arrayOfInt3[3] = R.drawable.bird_elite_injured_4;
duck_fly_double_2 = arrayOfInt3;
int[] arrayOfInt4 = new int[4];
arrayOfInt4[0] = R.drawable.bird_boss_1;
arrayOfInt4[1] = R.drawable.bird_boss_2;
arrayOfInt4[2] = R.drawable.bird_boss_3;
arrayOfInt4[3] = R.drawable.bird_boss_4;
duck_fly_three_1 = arrayOfInt4;
int[] arrayOfInt5 = new int[4];
arrayOfInt5[0] = R.drawable.bird_boss_block_1;
arrayOfInt5[1] = R.drawable.bird_boss_block_2;
arrayOfInt5[2] = R.drawable.bird_boss_block_3;
arrayOfInt5[3] = R.drawable.bird_boss_block_4;
duck_fly_three_block = arrayOfInt5;
int[] arrayOfInt6 = new int[4];
arrayOfInt6[0] = R.drawable.bird_boss_injured_1;
arrayOfInt6[1] = R.drawable.bird_boss_injured_2;
arrayOfInt6[2] = R.drawable.bird_boss_injured_3;
arrayOfInt6[3] = R.drawable.bird_boss_injured_4;
duck_fly_three_2 = arrayOfInt6;
nt[] arrayOfInt7 = new int[4];
rrayOfInt7[0] = R.drawable.bird_boss_injured_block_1;
rrayOfInt7[1] = R.drawable.bird_boss_injured_block_2;
rrayOfInt7[2] = R.drawable.bird_boss_injured_block_3;
rrayOfInt7[3] = R.drawable.bird_boss_injured_block_4;
uck_fly_three_2_block = arrayOfInt7;
t[] arrayOfInt8 = new int[4];
rrayOfInt8[0] = R.drawable.bird_boss_grave_injured_1;
rrayOfInt8[1] = R.drawable.bird_boss_grave_injured_2;
rrayOfInt8[2] = R.drawable.bird_boss_grave_injured_3;
rrayOfInt8[3] = R.drawable.bird_boss_grave_injured_4;
uck_fly_three_3 = arrayOfInt8;
t[] arrayOfInt9 = new int[6];
rayOfInt9[0] = R.drawable.bird_die_1;
rayOfInt9[1] = R.drawable.bird_die_1;
rayOfInt9[2] = R.drawable.bird_die_2;
rayOfInt9[3] = R.drawable.bird_die_2;
rayOfInt9[4] = R.drawable.bird_die_3;
rayOfInt9[5] = R.drawable.bird_die_3;
ck1_dead = arrayOfInt9;
t[] arrayOfInt10 = new int[6];
ayOfInt10[0] = R.drawable.bird2_die_1;
```

```
arrayOfInt10[1] = R.drawable.bird2_die_1;
arrayOfInt10[2] = R.drawable.bird2_die_2;
arrayOfInt10[3] = R.drawable.bird2_die_2;
arrayOfInt10[4] = R.drawable.bird2_die_3;
arrayOfInt10[5] = R.drawable.bird2_die_3;
duck2_dead = arrayOfInt10;
int[] arrayOfInt11 = new int[6];
arrayOfInt11[0] = R.drawable.bird3_die_1;
arrayOfInt11[1] = R.drawable.bird3_die_1;
arrayOfInt11[2] = R.drawable.bird3_die_2;
arrayOfInt11[3] = R.drawable.bird3_die_2;
arrayOfInt11[4] = R.drawable.bird3_die_3;
arrayOfInt11[5] = R.drawable.bird3_die_3;
duck3_dead = arrayOfInt11;
int[][] arrayOfInt = new int[3][];
arrayOfInt[0] = duck1_dead;
arrayOfInt[1] = duck2_dead;
arrayOfInt[2] = duck3_dead;
ducks_dead = arrayOfInt;


ıblic FlyDuck(Resources paramResources)

uper(paramResources);


ivate void getDrawable()

`(this.isshoot)

his.duckCount %= ducks_dead[(this.duck_type - 1)].length;
getDrawable(ducks_dead[(this.duck_type - 1)][this.duckCount]);
f (this.duckCount == ducks_dead[(this.duck_type - 1)].length - 1)
 this.orbit.setDisply(true);
his.duckCount = (1 + this.duckCount);


[ (this.duck_type == 1)

this.duckCount = (1 + this.duckCount);
this.duckCount %= duck_fly.length;
getDrawable(duck_fly[this.duckCount]);


(this.duck_type == 2)
```

```
    if (this.shootNumber == 0)
    {
     this.duckCount = (1 + this.duckCount);
     this.duckCount %= duck_fly_double_1.length;
     getDrawable(duck_fly_double_1[this.duckCount]);

    }
    this.duckCount = (1 + this.duckCount);
    this.duckCount %= duck_fly_double_2.length;
    getDrawable(duck_fly_double_2[this.duckCount]);

  }
  if (this.duck_type != 3)

  if (this.shootNumber == 0)
  {
   this.duckCount = (1 + this.duckCount);
   this.duckCount %= duck_fly_three_1.length;
   getDrawable(duck_fly_three_1[this.duckCount]);

  }
  if (this.shootNumber == 1)
  {
   if (this.isRight)
   {
    this.duckCount %= duck_fly_three_block.length;
    getDrawable(duck_fly_three_block[this.duckCount]);
    if (this.duckCount == duck_fly_three_block.length - 1)
     this.isRight = false;
    this.duckCount = (1 + this.duckCount);

   }
  this.duckCount = (1 + this.duckCount);
  this.duckCount %= duck_fly_three_2.length;
  getDrawable(duck_fly_three_2[this.duckCount]);

}
f (this.shootNumber != 2)

f (this.isRight)

this.duckCount %= duck_fly_three_2_block.length;
getDrawable(duck_fly_three_2_block[this.duckCount]);
if (this.duckCount == duck_fly_three_2_block.length - 1)
 this.isRight = false;
this.duckCount = (1 + this.duckCount);
```

```
    }
    this.duckCount = (1 + this.duckCount);
    this.duckCount %= duck_fly_three_3.length;
    getDrawable(duck_fly_three_3[this.duckCount]);

}

private void getDrawable(int paramInt)
{
 if (this.orbit.getXs() > 0.0D);
 for (this._currentDrawable = ResourceManager.getDrawable(this._resources, paramInt); ;
iis._currentDrawable = ResourceManager.getFlipDrawable(this._resources, paramInt))
  return;
}

)ublic void calcFrame(long paramLong)

long l = System.currentTimeMillis();
if (this.startTime == 0L);
for (this.startTime = l; ; this.startTime = l)
{
 if ((this.time >= 1000 * this.liveTime) && (!this.orbit.isTime()))
  this.orbit.setTime(true);
 if (this.stayTime <= 0L)
  this.orbit.setDisply(false);
this.orbit.calcFrame(paramLong);
this._x = this.orbit.getX();
this._y = this.orbit.getY();
getDrawable();
this._width = this._currentDrawable.getBitmap().getWidth();
this._height = this._currentDrawable.getBitmap().getHeight();
this.orbit.setWidthHeight(this._width, this._height);

this.stayTime -= l - this.startTime;
if (this.orbit.isDisply())
 continue;
this.time += l - this.startTime;


)lic void drawFrame(Canvas paramCanvas)

(!this.orbit.isDisply())

his.rec.set(0, 0, this._width, this._height);
```

```
setRect();
Paint localPaint = new Paint();
if (this.orbit.isTime())
{
 float[] arrayOfFloat = new float[20];
 arrayOfFloat[0] = 1.0F;
 arrayOfFloat[1] = 0.0F;
 arrayOfFloat[2] = 0.0F;
 arrayOfFloat[3] = 0.0F;
 arrayOfFloat[4] = 180.0F;
 arrayOfFloat[5] = 0.0F;
 arrayOfFloat[6] = 1.0F;
 arrayOfFloat[7] = 0.0F;
 arrayOfFloat[8] = 0.0F;
 arrayOfFloat[9] = 0.0F;
 arrayOfFloat[10] = 0.0F;
 arrayOfFloat[11] = 0.0F;
 arrayOfFloat[12] = 1.0F;
 arrayOfFloat[13] = 0.0F;
 arrayOfFloat[14] = 0.0F;
 arrayOfFloat[15] = 0.0F;
 arrayOfFloat[16] = 0.0F;
 arrayOfFloat[17] = 0.0F;
 arrayOfFloat[18] = 1.0F;
 arrayOfFloat[19] = 0.0F;
 localPaint.setColorFilter(new ColorMatrixColorFilter(arrayOfFloat));
}
paramCanvas.drawBitmap(this._currentDrawable.getBitmap(), this.rec, this.rect, localPaint);


olic void getFirstDrawable()

t i = 0;
(this.duck_type == 1)
 = duck_fly[this.duckCount];

his._currentDrawable = ResourceManager.getFlipDrawable(this._resources, i);
his._width = this._currentDrawable.getBitmap().getWidth();
is._height = this._currentDrawable.getBitmap().getHeight();
is.orbit.setWidthHeight(this._width, this._height);

(this.duck_type == 2)

i = duck_fly_double_1[this.duckCount];
```

```
    }
    if (this.duck_type != 3)
        i = duck_fly_three_1[this.duckCount];
}

public void setRectRectShoot()
{
 this.rectShoot.set(this._x - 35, this._y - 30, 35 + this._x, 30 + this._y);
}

ublic void setRectVoid()

 this.rect.set(this._x - this._width / 2, this._y - this._height / 2, this._x + this._width / 2, this._y
his._height / 2);
this._centerX = this.rect.centerX();
this._centerY = this.rect.centerY();


ublic void setUnRectVoid()
```

---

```
kage game.berburu.bebek.sprite;

ic class FlyOrbit
plements Orbit

vate double _curX;
vate double _curY;
vate GameMode _gm;
ate int _height;
ate double _leftX;
ate double _maxSpeed;
ate double _minSpeed;
ate double _range;
ate double _speedX;
te double _speedY;
te double _startX;
te int _width;
te double endX;
te GameLevel glv;
te boolean isDisply;
e boolean isTime;
```

```
private long nowTime = 0L;
private long pasuTime = 0L;
private double startY;
private double ys;

public FlyOrbit(int paramInt1, int paramInt2, double paramDouble1, double paramDouble2,
ameLevel paramGameLevel, GameMode paramGameMode)
{
this._width = paramInt1;
this._height = paramInt2;
this._minSpeed = paramDouble1;
this._maxSpeed = paramDouble2;
this._gm = paramGameMode;
this.glv = paramGameLevel;


rivate float getSpeedYByLevel(int paramInt)

loat f;
f (paramInt > 10)
f = 0.05F;

f = 0.05F * (paramInt - 1) / 10.0F;

rcturn f;


blic void calcFrame(long paramLong)

ng l1 = 0L;
ouble d = 0;

(this.nowTime == 0L)

his.nowTime = System.currentTimeMillis();
f (this.pasuTime > 0L)

if (l1 <= 0L)
l1 -= this.pasuTime;
this.pasuTime = 0L;

= Math.floor(l1 / 50L);
(this._leftX <= 0.1D)

f ((this._speedX < 0.0D) && (this._curX <= this.endX))
this.isDisply = true;
```

```
    if ((this._speedX > 0.0D) && (this._curX >= this.endX))
      this.isDisply = true;
    }
    if (!this.isDisply)
    {
     if (this._speedX >= 0.0D)

     if (this._curX >= this.endX)

    this._curX = this.endX;
    random();
    if (this._curY > this._height / 2)

    this.ys = 0.0D;
    this.ys = Math.min(this.ys + this._speedY, this._range);
   }
} else {

 this._curY += this.ys;

 long l2 = System.currentTimeMillis();
 l1 = l2 - this.nowTime;
this.nowTime = l2;
this.pasuTime = 0L;
this._curX += d * this._speedX;
if (!this.isTime)

his._leftX -= d * Math.abs(this._speedX);
f (this._speedX > 0.0D)

 if (this._curX > this.endX)
 {
  this._curX = this.endX;
  random();
}
this._curX += d * this._speedX;
if (!this.isTime)

this._leftX -= d * Math.abs(this._speedX);


is._curX += d * this._speedX;
(this._curY >= 240 - this._height / 2)

his.ys = 0.0D;
his.ys = Math.max(this.ys - this._speedY, -this._range);
```

```
    }
    if (this._curY > this.startY)
    {
     this.ys = Math.max(this.ys - this._speedY, -this._range);
    }
    this.ys = Math.min(this.ys + this._speedY, this._range);
  }
}

ublic double getSpeedByLevel(int paramInt, double paramDouble)

double d;
if (paramInt <= 10)
 d = paramDouble * (paramInt / 10.0F);
else
 d = paramDouble + (paramInt - 10) / 4.0F;
return d;


iblic int getX()

eturn (int)this._curX;


blic double getXs()

turn this._speedX;


lic int getY()

turn (int)this._curY;


lic int get_height()

urn this._height;


lic double get_startX()

rn this._startX;
```

```
public void init()
{
 this._curY = Tools.randomInt(this._height / 2, 240 - this._height / 2);
 if (Tools.randomInt(100) > 50)
 {
  this._curX = (-this._width);
  this._range = (4.2D + Tools.randomInt(20) / 10.0F);
  this._speedY = (0.1D + getSpeedYByLevel(this.glv.getLevel()));
  double d = this.glv.getMinlevel() / 10.0F;
  this._speedX = (getSpeedByLevel(this.glv.getLevel(), this._gm.getSpeedIncreaseStep()) + (d
Tools.randomDouble(this._minSpeed, this._maxSpeed)));
  this._leftX = (1000.0D * (5.0D * this._speedX) / 50.0D);
  if (this._curX < 240.0D)

  this._speedX = (-this._speedX);
  this.endX = (this._width / 2);
}
while (true)

 this._startX = this._curX;
 this.startY = this._curY;
 this.ys = this._range;



olic void initRandom()

(this._curX >= 240.0D)

his._speedX = (-Math.abs(this._speedX));
his.endX = (this._width / 2);

ile (true)


(this._curX < 240.0D)

his._speedX = Math.abs(this._speedX);
his.endX = (480 - this._width / 2);
ontinue;

urn;
```

```
public boolean isDisply()
{
 return this.isDisply;
}

public boolean isTime()
{
 return this.isTime;
}

ublic void random()

if (!this.isTime)
 initRandom();
while (true)
{
 this._startX = this._curX;
 this.nowTime = 0L;

 if (this._leftX > 480.0D)
 {
  initRandom();
  continue;
 }
 f (this._leftX > 480.0D)
  continue;
 esultRandom();
 eturn;


lic void resultRandom()

Math.abs(this._leftX - this._curX) < 20.0D)

is._speedX = (-Math.abs(this._speedX));
is.endX = (-this._width / 2);

ile (true)


(Math.abs(480.0D - this._leftX - this._curX) < 20.0D)

is._speedX = Math.abs(this._speedX);
is.endX = (480 + this._width / 2);
ntinue;
```

```
      }
      if (this._leftX > this._curX)
      {
       this._speedX = Math.abs(this._speedX);
       this.endX = ((this._leftX - this._curX) / 2.0D + this._curX);
       continue;
      }
      this._speedX = (-Math.abs(this._speedX));
      this.endX = (this._curX - (this._leftX + this._curX - 480.0D) / 2.0D);
      return;
    }
  }

  public void setDisply(boolean paramBoolean)
  {
   this.isDisply = paramBoolean;
  }

  public long setPauseTime(long paramLong)
  {
   this.pasuTime = paramLong;
   return 0L;
  }

  public void setTime(boolean paramBoolean)
  {
   this.isTime = paramBoolean;
  }

  public void setWidthHeight(int paramInt1, int paramInt2)
  {
   this._width = paramInt1;
   this._height = paramInt2;
  }

  public void setXs(double paramDouble)
  {
   this._speedX = paramDouble;


  public void set_height(int paramInt)

  this._height = paramInt;


  public void set_startX(double paramDouble)
```

```
  {
    this._startX = paramDouble;
  }
}
```

---

```
package game.berburu.bebek.sprite;

public class FlyOrbit
 implements Orbit
{
 private double _curX;
 private double _curY;
 private GameMode _gm;
 private int _height;
 private double _leftX;
 private double _maxSpeed;
 private double _minSpeed;
 private double _range;
 private double _speedX;
 private double _speedY;
 private double _startX;
 private int _width;
 private double endX;
 private GameLevel glv;
 private boolean isDisply;
 private boolean isTime;
 private long nowTime = 0L;
 private long pasuTime = 0L;
 private double startY;
 private double ys;

 public FlyOrbit(int paramInt1, int paramInt2, double paramDouble1, double paramDouble2,
GameLevel paramGameLevel, GameMode paramGameMode)
 {
   this._width = paramInt1;
   this._height = paramInt2;
   this._minSpeed = paramDouble1;
   this._maxSpeed = paramDouble2;
   this._gm = paramGameMode;
   this.glv = paramGameLevel;
 }

 private float getSpeedYByLevel(int paramInt)
 {
   float f;
```

```
    if (paramInt > 10)
      f = 0.05F;

      f = 0.05F * (paramInt - 1) / 10.0F;

      return f;
  }

  public void calcFrame(long paramLong)
  {
   long l1 = 0L;
   double d = 0;

   if (this.nowTime == 0L)
   {
    this.nowTime = System.currentTimeMillis();
    if (this.pasuTime > 0L)
    {
     if (l1 <= 0L)
     l1 -= this.pasuTime;
     this.pasuTime = 0L;
    }
    d = Math.floor(l1 / 50L);
    if (this._leftX <= 0.1D)
    {
     if ((this._speedX < 0.0D) && (this._curX <= this.endX))
       this.isDisply = true;
     if ((this._speedX > 0.0D) && (this._curX >= this.endX))
       this.isDisply = true;
    }
    if (!this.isDisply)
    {
     if (this._speedX >= 0.0D)

     if (this._curX >= this.endX)

     this._curX = this.endX;
     random();
     if (this._curY > this._height / 2)

     this.ys = 0.0D;
     this.ys = Math.min(this.ys + this._speedY, this._range);
    }
  } else {

   this._curY += this.ys;
```

```
}

public int getX()
{
 return (int)this._curX;
}

public double getXs()
{
 return this._speedX;
}

public int getY()
{
 return (int)this._curY;
}

public int get_height()
{
 return this._height;
}

public double get_startX()
{
 return this._startX;
}

public void init()
{
  this._curY = Tools.randomInt(this._height / 2, 240 - this._height / 2);
  if (Tools.randomInt(100) > 50)
  {
    this._curX = (-this._width);
    this._range = (4.2D + Tools.randomInt(20) / 10.0F);
    this._speedY = (0.1D + getSpeedYByLevel(this.glv.getLevel()));
    double d = this.glv.getMinlevel() / 10.0F;
    this._speedX = (getSpeedByLevel(this.glv.getLevel(), this._gm.getSpeedIncreaseStep()) + (d
+ Tools.randomDouble(this._minSpeed, this._maxSpeed)));
    this._leftX = (1000.0D * (5.0D * this._speedX) / 50.0D);
    if (this._curX < 240.0D)

    this._speedX = (-this._speedX);
    this.endX = (this._width / 2);
  }
  while (true)
  {
```

```
    this._startX = this._curX;
    this.startY = this._curY;
    this.ys = this._range;

  }
 }

 public void initRandom()
 {
  if (this._curX >= 240.0D)
  {
   this._speedX = (-Math.abs(this._speedX));
   this.endX = (this._width / 2);
  }
  while (true)
  {

   if (this._curX < 240.0D)
   {
    this._speedX = Math.abs(this._speedX);
    this.endX = (480 - this._width / 2);
    continue;
   }
   return;
 }


ublic boolean isDisply()

eturn this.isDisply;


ublic boolean isTime()

eturn this.isTime;


blic void random()

f (!this.isTime)
 initRandom();
vhile (true)

this._startX = this._curX;
this.nowTime = 0L;
```

```
    if (this._leftX > 480.0D)
    {
     initRandom();
     continue;
    }
    if (this._leftX > 480.0D)
     continue;
    resultRandom();
    return;
   }
  }

  public void resultRandom()
  {
   if (Math.abs(this._leftX - this._curX) < 20.0D)
   {
    this._speedX = (-Math.abs(this._speedX));
    this.endX = (-this._width / 2);
   }
   while (true)
   {

    if (Math.abs(480.0D - this._leftX - this._curX) < 20.0D)
    {
     this._speedX = Math.abs(this._speedX);
     this.endX = (480 + this._width / 2);
     continue;
    }
    if (this._leftX > this._curX)
    {
     this._speedX = Math.abs(this._speedX);
     this.endX = ((this._leftX - this._curX) / 2.0D + this._curX);
     continue;
    }
    this._speedX = (-Math.abs(this._speedX));
    this.endX = (this._curX - (this._leftX + this._curX - 480.0D) / 2.0D);
    return;
   }
  }

  public void setDisply(boolean paramBoolean)
  {
   this.isDisply = paramBoolean;
  }

  public long setPauseTime(long paramLong)
```

```
{
 this.pasuTime = paramLong;
 return 0L;
}

public void setTime(boolean paramBoolean)
{
 this.isTime = paramBoolean;
}

public void setWidthHeight(int paramInt1, int paramInt2)
{
 this._width = paramInt1;
 this._height = paramInt2;
}

public void setXs(double paramDouble)
{
 this._speedX = paramDouble;
}

public void set_height(int paramInt)
{
 this._height = paramInt;
}

public void set_startX(double paramDouble)
{
 this._startX = paramDouble;
}
}
```

---

```
package game.berburu.bebek.sprite;

public class GameLevel
{
 private int hitNumber;
 private int level;
 private int minlevel;
 private int money;

 public int getHitNumber()
 {
  return this.hitNumber;
```

```
}

public int getLevel()
{
 return this.level;
}

public int getMinlevel()
{
 return this.minlevel;
}

public int getMoney()
{
 return this.money;
}

public void setHitNumber(int paramInt)
{
 this.hitNumber = paramInt;
}

public void setLevel(int paramInt)
{
 this.level = paramInt;
}

public void setMinlevel(int paramInt)
{
 this.minlevel = paramInt;
}

public void setMoney(int paramInt)
{
 this.money = paramInt;
}
}
```

---

```
package game.berburu.bebek.sprite;

public class GameMode
{

 public double fly_maxspeed, fly_minspeed, speedIncreaseStep;
 public static final GameMode EASY =null;
```

```
 public static final GameMode MIDDLE =null  ;
 public static final GameMode HARD  ;

 static {
        HARD = new GameMode("HARD", 2, 6.0D, 7.0D, 3.0D);
        GameMode[] arrayOfGameMode = new GameMode[3];
  arrayOfGameMode[0] = EASY;
  arrayOfGameMode[1] = MIDDLE;
  arrayOfGameMode[2] = HARD;
//
 }

 private GameMode(double paramDouble2, double paramDouble3, double arg7)
 {
  this.fly_minspeed = paramDouble2;
  this.fly_maxspeed = paramDouble3;
  this.speedIncreaseStep = arg7;
 }

 public GameMode(String string, int i, double d, double e, double f) {
        // TODO Auto-generated constructor stub
}

public double getFly_maxspeed()
 {
  return this.fly_maxspeed;
 }

 public double getFly_minspeed()
 {
  return this.fly_minspeed;
 }

 public double getSpeedIncreaseStep()
 {
  return this.speedIncreaseStep;
 }
}
```

---

```
package game.berburu.bebek.sprite;


import game.berburu.bebek.R;
import android.content.res.Resources;
import android.graphics.Bitmap;
```

```
mport android.graphics.BitmapFactory;

ublic class Gun
{
 public static final Gun GUN1 = new Gun(3, GunReload.Long, GunSight.Small,
GunRecoil.Max, R.drawable.gun1_normal, R.drawable.gun1_highlight,
R.drawable.gun1_locked, R.drawable.gun1_game, 0);
 public static final Gun GUN2 = new Gun(5, GunReload.Medium, GunSight.Small,
GunRecoil.Medium, R.drawable.gun2_normal,R.drawable.gun2_highlight,
R.drawable.gun2_locked, R.drawable.gun2_game,5000);
 public static final Gun GUN3 = new Gun(7, GunReload.Medium, GunSight.Medium,
GunRecoil.Medium, R.drawable.gun3_normal,R.drawable.gun3_highlight,
R.drawable.gun3_locked, R.drawable.gun3_game, 20000);
 public static final Gun GUN4 = new Gun(7, GunReload.Short, GunSight.Medium,
GunRecoil.Mini, R.drawable.gun4_normal,R.drawable.gun4_highlight,
R.drawable.gun4_locked, R.drawable.gun4_game,60000);
 public static final Gun GUN5 = new Gun(10, GunReload.Instant, GunSight.Large,
GunRecoil.None, R.drawable.gun5_normal,R.drawable.gun5_highlight,
R.drawable.gun5_locked, R.drawable.gun5_game,120000);
 private final int _ballnumer;
 private final int _costMoney;
 private final int _gamgunid;
 private final int _gunBmpId;
 private final GunSight _gunSight;
 private final int _highlightedId;
 private final int _lockedId;
 private final GunRecoil _recoil;
 private final GunReload _reloadTime;

 private Gun(int paramInt1, GunReload paramGunReload, GunSight paramGunSight,
GunRecoil paramGunRecoil, int paramInt2, int paramInt3, int paramInt4, int paramInt5, int
paramInt6)
 {
  this._ballnumer = paramInt1;
  this._reloadTime = paramGunReload;
  this._gunSight = paramGunSight;
  this._recoil = paramGunRecoil;
  this._costMoney = paramInt6;
  this._gunBmpId = paramInt2;
  this._highlightedId = paramInt3;
  this._lockedId = paramInt4;
  this._gamgunid = paramInt5;
 }

 public int getBullets()
 {
```

```
 return this._ballnumer;
}

public int getCostMoney()
{
 return this._costMoney;
}

public int getGamGunId()
{
 return this._gamgunid;
}

public int getGunBmpId()
{
 return this._gunBmpId;
}

public int getHighlightedId()
{
 return this._highlightedId;
}

public int getLockedId()
{
 return this._lockedId;
}

public GunRecoil getRecoil()
{
 return this._recoil;
}

public GunReload getReload()
{
 return this._reloadTime;
}

public GunSight getSight()
{
 return this._gunSight;
}

public static enum GunRecoil
{
        Max
```

```
      , Medium
      , Mini, None ;
      private int _descStrId;
private int _value;

private GunRecoil() {
            // TODO Auto-generated constructor stub
      }

static
{
 GunRecoil[] arrayOfGunRecoil = new GunRecoil[4];
 arrayOfGunRecoil[0] = Max;
 arrayOfGunRecoil[1] = Medium;
 arrayOfGunRecoil[2] = Mini;
 arrayOfGunRecoil[3] = None;
 //ret arrayOfGunRecoil;
}

private GunRecoil(int arg3, int arg4)
{
 int j = 0;
 this._descStrId = j;
 int i = 0;
 this._value = i;
}

public String getDesc(Resources paramResources)
{
 return paramResources.getString(this._descStrId);
}

public int getValue()
{
 return this._value;
}
}

public static class GunReload
{
      public static final GunReload Long=null;
      public static final GunReload Medium=null;
      public static final GunReload Short=null;
      public static final GunReload Instant;
      private int _reloadMs;
```

```
  static
  {
   Instant = new GunReload("Instant", 3, 0);
   GunReload[] arrayOfGunReload = new GunReload[4];
   arrayOfGunReload[0] = Long;
   arrayOfGunReload[1] = Medium;
   arrayOfGunReload[2] = Short;
   arrayOfGunReload[3] = Instant;
  // re arrayOfGunReload;
  }

  private GunReload(int arg3)
  {

   this._reloadMs = arg3;
  }

  public GunReload(String string, int i, int j) {
                // TODO Auto-generated constructor stub
        }

        public String getDesc(Resources paramResources)
  {
   Object[] arrayOfObject = null;
   if (this._reloadMs != 0)
   {
    arrayOfObject = new Object[1];
    arrayOfObject[0] = this._reloadMs;
   }
   for (String str = paramResources.getString(R.string.desc_reload_time, arrayOfObject); ; str =
paramResources.getString(R.string.desc_reload_time_instant))
    return str;
  }

  public int getValue()
  {
   return this._reloadMs;
  }
 }

 public static class GunSight
 {

        private static final GunSight Large = null;
        private static final GunSight Medium = null;
        private static final GunSight Small = null;
```

```
    private final int _collisionSize;
  private final int _descStringId;
  private Bitmap _drawable;
  private final int _drawableId;

  static
  {
   GunSight[] arrayOfGunSight = new GunSight[3];
   arrayOfGunSight[0] = Large;
   arrayOfGunSight[1] = Medium;
   arrayOfGunSight[2] = Small;
  // GunSight arrayOfGunSight1;
  }

 private GunSight(int paramInt3, int arg4, int arg5)
 {
  this._descStringId = paramInt3;
  int i = 0;
  this._drawableId = i;
  int j = 0;
  this._collisionSize = j;


 ublic int getCollisionSize()

 return this._collisionSize;


 ublic String getDesc(Resources paramResources)

 eturn paramResources.getString(this._descStringId);


 blic Bitmap getDrawable(Resources paramResources)

 f (this._drawable == null)
 his._drawable = BitmapFactory.decodeResource(paramResources, this._drawableId);
 turn this._drawable;
```

---

```
ge game.berburu.bebek.sprite;

t game.berburu.bebek.R;
```

```
mport game.berburu.bebek.ResourceManager;
mport android.content.res.Resources;
mport android.graphics.Bitmap;
mport android.graphics.Canvas;
mport android.graphics.drawable.BitmapDrawable;


ublic class Life extends AbstractSprite
{
 private int _num;
 private BitmapDrawable imageDrable = ResourceManager.getDrawable(this._resources,
R.drawable.life);

 public Life(Resources paramResources)
 {
  super(paramResources);
  this._width = this.imageDrable.getBitmap().getWidth();
  this._height = this.imageDrable.getBitmap().getHeight();
  this._x = 25;
  this._y = 10;
 }

 public void calcFrame(long paramLong)
 {
 }

 public void drawFrame(Canvas paramCanvas)
 {
  this._x = 25;
  for (int i = 0; ; i++)
  {
   if (i >= this._num)
    return;
   setRectVoid();
   this.imageDrable.setBounds(this.rect);
   this.imageDrable.draw(paramCanvas);
   this._x = (this.imageDrable.getBitmap().getWidth() + this._x);
  }
 }

 public int get_num()
 {
  return this._num;
 }

 public void setRectRectShoot()
```

```
 {
 }

 public void setUnRectVoid()
 {
 }

 public void set_num(int paramInt)
 {
  this._num = paramInt;
 }
}
```

---

```
package game.berburu.bebek.sprite;

public abstract interface Orbit
{
 public abstract void calcFrame(long paramLong);

 public abstract int getX();

 public abstract double getXs();

 public abstract int getY();

 public abstract boolean isDisply();

 public abstract boolean isTime();

 public abstract void setDisply(boolean paramBoolean);

 public abstract long setPauseTime(long paramLong);

 public abstract void setTime(boolean paramBoolean);

 public abstract void setWidthHeight(int paramInt1, int paramInt2);

 public abstract void setXs(double paramDouble);
}
```

---

```
package game.berburu.bebek.sprite;

import java.io.PrintStream;
import java.util.Random;
```

```
public class Tools
{
 private static Random _random = new Random();

 public static void main(String[] paramArrayOfString)
 {
  System.out.print(randomInt(1, 2));
 }

 public static double randomDouble(double paramDouble1, double paramDouble2)
 {
  double d = paramDouble2 * _random.nextDouble();
  while (true)
  {
   if (d >= paramDouble1)
    return d;
   d = paramDouble2 * _random.nextDouble();
  }
 }

 public static int randomInt(int paramInt)
 {
  return _random.nextInt(paramInt);
 }

 public static int randomInt(int paramInt1, int paramInt2)
 {
  return paramInt1 + _random.nextInt(paramInt2 - paramInt1);
 }
}
```

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

# FORMULIR BIMBINGAN SKRIPSI

Nama : DAVID JOAN PERMANA
Nim : 06.12.507
Masa Bimbingan : 07 Januari 2012 – 07 Juni 2012
Judul Skripsi : Rancang Bangun Aplikasi Game Duck Hunter Berbasis Android

| No. | Tanggal | Uraian | Parap Pembimbing |
|---|---|---|---|
| 1. | 16-06-2012 | Bab 1 dan 2 ACC | |
| 2. | 23-06-2012 | Revisi Bab 3, perbaikan flowchart | |
| 3. | 24-06-2012 | Bab 3 ACC | |
| 4. | 25-06-2012 | Bab 4 Revisi, menambah tabel uji coba aplikasi dan tabel perbandingan aplikasi. | |
| 5. | 26-06-2012 | Bab 4 ACC | |
| 6. | 28-06-2012 | Bab 5 ACC | |
| 7. | | | |
| 8. | | | |
| 9. | | | |
| 10. | | | |

Malang,
Dosen Pembimbing I

**Sandy Nataly Mantja, S.KOM**
**NIP.P 103.0800.418**

Form.S-4b

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

# FORMULIR BIMBINGAN SKRIPSI

Nama : DAVID JOAN PERMANA
Nim : 06.12.507
Masa Bimbingan : 07 Januari 2012 – 07 Juni 2012
Judul Skripsi : Rancang Bangun Aplikasi Game Duck Hunter Berbasis Android

| No. | Tanggal | Uraian | Parap Pembimbing |
|---|---|---|---|
| 1. | 16-06-2012 | Revisi bab 1 dan 2, perbaikan spasi, huruf besar, dan perletakan gambar. | |
| 2. | 23-06-2012 | Bab 1 dab 2 ACC. | |
| 3. | 23-06-2012 | Revisi bab 3, perbaikan alinea dan judul gambar. | |
| 4. | 23-06-2012 | Bab 3 dan 4 ACC. | |
| 5. | 25-06-2012 | Revisi Bab 5, menambah daftar pustaka. | |
| 6. | 28-06-2012 | Bab 5 ACC. | |
| 7. | | | |
| 8. | | | |
| 9. | | | |
| 10. | | | |

Malang,
Dosen Pembimbing II

**Ibrahim Ashari, ST, MT**
**NIP.Y. 1018700.151**

Form.S-4b

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
PROGRAM STUDI TEKNIK ELEKTRO S-1
Jl. Karanglo KM. 2 Malang

# FORMULIR PERBAIKAN UJIAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata 1 Program Studi Teknik Elektro Konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : DAVID JOAN PERMANA
NIM : 06.12.507
PROGRAM STUDI : Teknik Komputer dan Informatika S-1
JUDUL : **RANCANG BANGUN APLIKASI *GAME* DUCK HUNTER BERBASIS ANDROID**

| No | Penguji | Tanggal | Uraian | Paraf |
|---|---|---|---|---|
| 1. | Penguji I | 31 Juli 2012 | 1. Perbaikan semua table<br>2. Perbaikan format penulisan | |
| 2. | Penguji II | 31 Juli 2012 | 1. Perbaikan format penulisan | |

**Disetujui :**

**Dosen Penguji I**

**Aryuanto Soetedjo, Dr.Eng. ST., MT.**
**NIP.Y. 1030800417**

**Dosen Penguji II**

**Bambang Prio Hartono, ST, MT.**
**NIP.Y.1028400082**

**Mengetahui :**

**Dosen Pembimbing I**

**Sandy Nataly Manntja, S.KOM.**
**NIP.P. 1030800418**

**Dosen Pembimbing II**

**Ibrahim Ashari, ST, MT.**
**NIP.Y.1018700151**