RANCANG BANGUN GAME STRATEGI SPONGEBOB DENGAN BAHASA PEMROGRAMAN JAVA SCRIPT MENGGUNAKAN NETBEAN IDE 6.0

SKRIPSI

Disusun Oleh :

ACHMAD TAUFIQ

07.12.546

PROGRAM STUDI TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

2013

LEMBAR PERSETUJUAN

RANCANG BANGUN GAME STRATEGI SPONGEBOB DENGAN BAHASA PEMROGRAMAN JAVA SCRIPT MENGGUNAKAN NETBEAN IDE 6.0

SKRIPSI

Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh

Gelar Sarjana Teknik Strata Satu (S-1)

Disusun oleh :

ACHMAD TAUFIQ

NIM : 07. 12. 546

Mengetahui,

Ketua Jurusan Teknik Elektro S-1

M. Ibrahim Ashari, ST,MT
NIP.P.1030100358

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
M. Ibrahim Ashari ST,MT	**Sandy NatalMantja,Skom**
NIP. P. 1030100358	**NIP. P. 1030800418**

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2013

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini :

Nama : ACHMAD TAUFIQ

NIM : 07.12.546

Program Studi : Teknik Elektro S-1

Konsentrasi : Teknik Komputer dan Informatika

Dengan ini menyatakan bahwa Skripsi yang saya buat adalah hasil karya sendiri, tidak merupakan plagiasi dari karya orang lain. Dalam Skripsi ini tidak memuat karya orang lain, kecuali dicantumkan sumbernya sesuai dengan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat, dan apabila di kemudian hari ada pelanggaran atas surat pernyataan ini, saya bersedia menerima sangsinya.

Malang, 16 April 2013

Yang membuat Pernyataan,

Achmad Taufiq
NIM. 07.12.546

Abstrak

***SpongeBob SquarePants**, tokoh utama dalam kartun ini yang adalah seekor spons yang sebenarnya berbentuk spons mandi berwarna kuning ini adalah pribadi yang baik, mudah diajak berteman, dan optimistis. Spongebob tinggal di dalam rumah berbentuk nanas di laut, di Jalan Conch nomor 124, Bikini Bottom. Dia juga memelihara seekor siput yang bernama Gary. Pekerjaannya sehari- hari adalah koki di rumah makan Krusty Krab (dia sendiri pun mendapat penghargaan "Employee of the Month" (Pegawai Teladan Bulan Ini) 374 kali berturut - turut), yang terkenal dengan burgernya Krabby Patty. Dia juga bersekolah di Mrs. Puff Boating School, sekolah mengemudi Mrs. Puff, namun selalu gagal ketika tes mengemudi .*

Java adalah satu set produk perangkat lunak komputer beberapa spesifikasi dari Sun Microsystems (yang telah bergabung dengan Oracle Corporation), yang bersama-sama menyediakan sistem untuk mengembangkan perangkat lunak aplikasi dan penggelaran di lingkungan komputasi cross-platform.

Kata kunci :Komputer, Game sponge bob, Netbeans, Java

Abstract

SpongeBob SquarePants, the main character in this cartoon is actually a sponge shaped yellow sponge bath is a good person, easy to get along with, and optimistic. Spongebob live in a pineapple-shaped home in the sea, at number 124 Conch Street, Bikini Bottom. He also maintains a snail named Gary. Their daily work is the chef at the restaurant Krusty Krab (she was awarded "Employee of the Month" (Exemplary Employee Month) 374 times in a row - co) the famous Krabby Patty burgers. He also attended Mrs. Puff Boating School, Mrs. Driving school. Puff, but failed when test driving.

Java is a set of computer software products multiple specifications from Sun Microsystems (which has merged with Oracle Corporation), which together provide a system for developing application software and deploying it in a cross-platform computing environment.

Keywords: Computers, Games Sponge Bob Netbeans, Java

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa, yang telah memberikan berkat-Nya, sehingga penulis dapat menyelesaikan laporan Skripsi ini dengan baik dan lancar.

Laporan Skripsi ini merupakan salah satu persyaratan akademik dalam menyelesaikan program Strata 1 Jurusan Teknik Elektro, Konsentrasi Komputer & Informatika, Institut Teknologi Nasional Malang. Adapun judul laporan Skripsi ini adalah:

RANCANG BANGUN GAME STRATEGI SPONGEBOB DENGAN BAHASA PEMROGRAMAN JAVA SCRIPT MENGGUNAKAN NETBEAN IDE 6.0

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 M. Ibrahim Ashari, ST, 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. Bapak M. Ibrahim Ashari, ST, MT selaku Dosen Pembimbing I
4. Ibu Sandy Natalay Manjta, S.kom selaku Dosen Pembimbing II
5. Orang Tua, dan kedua kakak ku yang telah memberikan dukungan untuk selalu berdoa, berusaha dan nasehat yang telah diberikan sampai saat ini.
6. Seluruh dosen dan pegawai ITN Kampus 2 Malang.
7. Semua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini yang tidak bisa penulis sebutkan satu persatu.

Penulis berharap agar buku laporan Skripsi ini dapat memberikan banyak manfaat bagi semua pihak yang membutuhkan, khususnya bagi rekan-rekan mahasiswa. Penulis menyadari bahwa dalam penyusunan laporan ini masih banyak kekurangan, oleh karena itu mohon maaf apabila dalam buku ini terdapat hal-hal yang kurang berkenan dihati para pembaca.

Penulis juga mengharap koreksi, kritik serta saran-saran yang bermanfaat demi kesempurnaan buku Laporan Skripsi ini.

Malang, 15 April 2013

Penulis

DAFTAR ISI

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1. PENDAHULUAN

1.1. LATAR BELAKANG

Dimasa sekarang game selain telah menjadi kebutuhan bagi setiap orang untuk melepaskan rasa jenuh dan memberi perasaan yang lebih santai, juga dapat mengajarkan orang cara pikir yang lebih baik dan kritis.

Pesatnya perkembangan game dibuktikan dengan semakin menjamurnya industri game dan semakin banyaknya aplikasi game yang dihasilkan untuk berbagai macam platform. Tipe permainan dalam sebuah game pun semakin beragam dan memiliki keunikan tersendiri sehingga menarik rasa penasaran dari para pemain untuk menyelesaikan game tersebut. Setiap jenis game memiliki kelebihan dan kelemahan tersendiri.

Java adalah bahasa pemograman yang bdapat di jalankan di berbagai komputer termasuk telpon genggam . bahasa ini banyak mengadopsi sintaksis yang terdapat pada C da C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin – rutin atas bawah yang minimal , saat ini java merupakan bahasa pemograman yang paling popular digunakan , dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi game maupun berbasis web

1.2. RUMUSAN MASALAH

Berdasarkan uraian latar belakang diatas maka rumusan masalahnya adalah Bagaimana cara membuat game strategi spongebob dengan menggunakan bahasa pemrograman java script menggunakan Netbeans 6.0

1.3. TUJUAN

Tujuan umum dari penulisan laporan Tugas Akhir ini adalah Bagaimana cara mendesain dan membuat sebuah game Spongebob Fighter yang menggunakan perangkat lunak Netbeans IDE 6.0

1.4. BATASAN MASALAH

Agar diperoleh persamaan persepsi, maka dalam penyusunan laporan ini hanya dibatasi pada beberapa hal :

1. Mempunyai dua pilihan setting tempat untuk bermain.
2. Game dimainkan hanya untuk 2 pemain (komputer dan user).
3. Game hanya bisa dimainkan secara offline.
4. Tingkat kesulitan permainan game ini maksimal sampai level 3.

1.5. .METODE PENELETIAN

Metode penelitian yang di terapkan dalam penulisan ini tugas akhir ini meliputi :

1. Analisa Kebutuhan Sistem

Data dan informasi yang telah diperoleh akan dianalisa agar didapatkan kerangka global yang bertujuan untuk mendefinisikan kebutuhan system dimana nantinya akan digunakan sebagai acuan perancangan system

2. Perancangan dan pembuatan

Berdasarkan data dan informasi yang telah diperoleh serta analisa kebutuhan untuk membangun system ini , akan dibuat rancangan kerangka global yang menggambarkan mekanisme dari system yang akan dibuat dan diimplementasikan kedalam system

3. Pengujian dan Evaluasi

Pada tahap ini , system yang telah selesai di buat akan diuji coba , yaitu pengujian berdasarkan fungsionalitas program dan akan dilakukan koreksi dan penyempurnaan program jika diperlukan

1.6 SISTEMATIKA PENULISAN

Dalam penyusunan laporan Tugas Akhir ini akan diuraikan menjadi beberapa bab, dan masing-masing bab akan dipaparkan dalam beberapa sub bab, diantaranya :

BAB I. Pendahuluan

Dalam bab ini akan menjelaskan latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan laporan Skripsi

BAB II. Landasan Teori

Dalam bab ini akan membahas dan menjelaskan mengenai dasar teoritis yang menjadi landasan dan mendukung pelaksanaan penulisan laporan Tugas Akhir.

BAB III. Perancangan Game

Dalam bab ini menyampaikan informasi tentang pembuatan *game* dari awal hingga akhir.

BAB IV. Implementasi Game

Dalam bab ini akan membahas pengujian aplikasi perancangan *game* menggunakan Java dan *software* pendukung Netbeans 6.0 IDE.

BAB V. Penutup

Dalam bab ini akan disampaikan kesimpulan dan saran dari keseluruhan bahasan.

BAB I

PENDAHULUAN

1. PENDAHULUAN

1.1. LATAR BELAKANG

Dimasa sekarang game selain telah menjadi kebutuhan bagi setiap orang untuk melepaskan rasa jenuh dan memberi perasaan yang lebih santai, juga dapat mengajarkan orang cara pikir yang lebih baik dan kritis.

Pesatnya perkembangan game dibuktikan dengan semakin menjamurnya industri game dan semakin banyaknya aplikasi game yang dihasilkan untuk berbagai macam platform. Tipe permainan dalam sebuah game pun semakin beragam dan memiliki keunikan tersendiri sehingga menarik rasa penasaran dari para pemain untuk menyelesaikan game tersebut. Setiap jenis game memiliki kelebihan dan kelemahan tersendiri.

Java adalah bahasa pemograman yang bdapat di jalankan di berbagai komputer termasuk telpon genggam . bahasa ini banyak mengadopsi sintaksis yang terdapat pada C da C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin – rutin atas bawah yang minimal , saat ini java merupakan bahasa pemograman yang paling popular digunakan , dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi game maupun berbasis web

1.2. RUMUSAN MASALAH

Berdasarkan uraian latar belakang diatas maka rumusan masalahnya adalah Bagaimana cara membuat game strategi spongebob dengan menggunakan bahasa pemrograman java script menggunakan Netbeans 6.0

1.3. TUJUAN

Tujuan umum dari penulisan laporan Tugas Akhir ini adalah Bagaimana cara mendesain dan membuat sebuah game Spongebob Fighter yang menggunakan perangkat lunak Netbeans IDE 6.0

1.4. BATASAN MASALAH

Agar diperoleh persamaan persepsi, maka dalam penyusunan laporan ini hanya dibatasi pada beberapa hal :

5. Mempunyai dua pilihan setting tempat untuk bermain.
6. Game dimainkan hanya untuk 2 pemain (komputer dan user).
7. Game hanya bisa dimainkan secara offline.
8. Tingkat kesulitan permainan game ini maksimal sampai level 3.

1.5. .METODE PENELETIAN

Metode penelitian yang di terapkan dalam penulisan ini tugas akhir ini meliputi :

1. Analisa Kebutuhan Sistem

Data dan informasi yang telah diperoleh akan dianalisa agar didapatkan kerangka global yang bertujuan untuk mendefinisikan kebutuhan system dimana nantinya akan digunakan sebagai acuan perancangan system

4. Perancangan dan pembuatan

Berdasarkan data dan informasi yang telah diperoleh serta analisa kebutuhan untuk membangun system ini , akan dibuat rancangan kerangka global yang menggambarkan mekanisme dari system yang akan dibuat dan diimplementasikan kedalam system

5. Pengujian dan Evaluasi

Pada tahap ini , system yang telah selesai di buat akan diuji coba , yaitu pengujian berdasarkan fungsionalitas program dan akan dilakukan koreksi dan penyempurnaan program jika diperlukan

1.7 SISTEMATIKA PENULISAN

Dalam penyusunan laporan Tugas Akhir ini akan diuraikan menjadi beberapa bab, dan masing-masing bab akan dipaparkan dalam beberapa sub bab, diantaranya :

BAB I. Pendahuluan

Dalam bab ini akan menjelaskan latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan laporan Skripsi

BAB II. Landasan Teori

Dalam bab ini akan membahas dan menjelaskan mengenai dasar teoritis yang menjadi landasan dan mendukung pelaksanaan penulisan laporan Tugas Akhir.

BAB III. Perancangan Game

Dalam bab ini menyampaikan informasi tentang pembuatan *game* dari awal hingga akhir.

BAB IV. Implementasi Game

Dalam bab ini akan membahas pengujian aplikasi perancangan *game* menggunakan Java dan *software* pendukung Netbeans 6.0 IDE.

BAB V. Penutup

Dalam bab ini akan disampaikan kesimpulan dan saran dari keseluruhan bahasan.

BAB I

PENDAHULUAN

1. PENDAHULUAN

1.1. LATAR BELAKANG

Dimasa sekarang game selain telah menjadi kebutuhan bagi setiap orang untuk melepaskan rasa jenuh dan memberi perasaan yang lebih santai, juga dapat mengajarkan orang cara pikir yang lebih baik dan kritis.

Pesatnya perkembangan game dibuktikan dengan semakin menjamurnya industri game dan semakin banyaknya aplikasi game yang dihasilkan untuk berbagai macam platform. Tipe permainan dalam sebuah game pun semakin beragam dan memiliki keunikan tersendiri sehingga menarik rasa penasaran dari para pemain untuk menyelesaikan game tersebut. Setiap jenis game memiliki kelebihan dan kelemahan tersendiri.

Java adalah bahasa pemograman yang bdapat di jalankan di berbagai komputer termasuk telpon genggam . bahasa ini banyak mengadopsi sintaksis yang terdapat pada C da C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin – rutin atas bawah yang minimal , saat ini java merupakan bahasa pemograman yang paling popular digunakan , dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi game maupun berbasis web

1.2. RUMUSAN MASALAH

Berdasarkan uraian latar belakang diatas maka rumusan masalahnya adalah Bagaimana cara membuat game strategi spongebob dengan menggunakan bahasa pemrograman java script menggunakan Netbeans 6.0

1.3. TUJUAN

Tujuan umum dari penulisan laporan Tugas Akhir ini adalah Bagaimana cara mendesain dan membuat sebuah game Spongebob Fighter yang menggunakan perangkat lunak Netbeans IDE 6.0

1.4. BATASAN MASALAH

Agar diperoleh persamaan persepsi, maka dalam penyusunan laporan ini hanya dibatasi pada beberapa hal :

9. Mempunyai dua pilihan setting tempat untuk bermain.
10. Game dimainkan hanya untuk 2 pemain (komputer dan user).
11. Game hanya bisa dimainkan secara offline.
12. Tingkat kesulitan permainan game ini maksimal sampai level 3.

1.5. .METODE PENELETIAN

Metode penelitian yang di terapkan dalam penulisan ini tugas akhir ini meliputi :

1. Analisa Kebutuhan Sistem

Data dan informasi yang telah diperoleh akan dianalisa agar didapatkan kerangka global yang bertujuan untuk mendefinisikan kebutuhan system dimana nantinya akan digunakan sebagai acuan perancangan system

6. Perancangan dan pembuatan

Berdasarkan data dan informasi yang telah diperoleh serta analisa kebutuhan untuk membangun system ini , akan dibuat rancangan kerangka global yang menggambarkan mekanisme dari system yang akan dibuat dan diimplementasikan kedalam system

7. Pengujian dan Evaluasi

Pada tahap ini , system yang telah selesai di buat akan diuji coba , yaitu pengujian berdasarkan fungsionalitas program dan akan dilakukan koreksi dan penyempurnaan program jika diperlukan

1.8 SISTEMATIKA PENULISAN

Dalam penyusunan laporan Tugas Akhir ini akan diuraikan menjadi beberapa bab, dan masing-masing bab akan dipaparkan dalam beberapa sub bab, diantaranya :

BAB I. Pendahuluan

Dalam bab ini akan menjelaskan latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan laporan Skripsi

BAB II. Landasan Teori

Dalam bab ini akan membahas dan menjelaskan mengenai dasar teoritis yang menjadi landasan dan mendukung pelaksanaan penulisan laporan Tugas Akhir.

BAB III. Perancangan Game

Dalam bab ini menyampaikan informasi tentang pembuatan *game* dari awal hingga akhir.

BAB IV. Implementasi Game

Dalam bab ini akan membahas pengujian aplikasi perancangan *game* menggunakan Java dan *software* pendukung Netbeans 6.0 IDE.

BAB V. Penutup

Dalam bab ini akan disampaikan kesimpulan dan saran dari keseluruhan bahasan.

BAB II

LANDASAN TEORI

2.1 Ide cerita dalam pembuatan game

2.1.2 Karakter

a. Spongebob Squarepants

SpongeBob SquarePants, tokoh utama dalam kartun ini yang adalah seekor spons yang sebenarnya berbentuk spons mandi berwarna kuning ini adalah pribadi yang baik, mudah diajak berteman, dan optimistis. Spongebob tinggal di dalam rumah berbentuk nanas di laut, di Jalan Conch nomor 124, Bikini Bottom. Dia juga memelihara seekor siput yang bernama Gary. Pekerjaannya sehari- hari adalah koki di rumah makan Krusty Krab (dia sendiri pun mendapat penghargaan *"Employee of the Month"* (Pegawai Teladan Bulan Ini) 374 kali berturut - turut), yang terkenal dengan burgernya Krabby Patty. Dia juga bersekolah di Mrs. Puff Boating School, sekolah mengemudi Mrs. Puff, namun selalu gagal ketika tes mengemudi. Ia juga suka berburu ubur ubur.

Gambar 2.1. Spongebob Squarepants

b. Patrick Star

Patrick Star adalah seekor bintang laut berwarna merah muda yang merupakan sahabat Spongebob. Walaupun sering kali tidak berkonsentrasi dalam percakapannya dan terjadi salah pengertian, Patrick adalah sahabat yang baik. Patrick tinggal bersebelahan dengan rumah Squidward yang juga bersebelahan dengan rumah nanas Spongebob dan rumahnya terletak di bawah batu. Patrick merupakan hewan paling bodoh di Bikini Bottom karena dia sering melakukan perbuatan yang konyol dan bodoh.

Patrick tidak pergi ke sekolah kecuali ke sekolah mengemudi (hanya sekali saja dalam episode New Student Starfish karena menemani SpongeBob). Patrick juga tidak bekerja, segala perbelanjaan untuk kehidupannya di Bikini Bottom selalu diberikan oleh ibu bapaknya (Herb dan Margie) yang menetap jauh dari Bikini Bottom.

Setiap pagi Patrick akan berdiri di luar rumahnya untuk berkata "Hai SpongeBob" kepada SpongeBob yang akan pergi ke tempat kerjanya di Krusty Krab dan Patrick akan terus berdiri di luar rumahnya sampai SpongeBob pulang dari tempat kerjanya untuk bermain berdua. Patrick dan SpongeBob selalu mengganggu tetangga mereka Squidward Tentacles (meskipun mereka berdua tidak menganggap yang mereka lakukan itu menggangu.

Gambar 2.2. Patrick Star

c . Squidward Tentacles

Squidward Tentacles adalah seorang dewasa berusia 32 tahun, ia kasir yang sering membuat ulah,dan membuat repot semua orang di sekitarnya. Ia tidak senang dengan kedua tetangganya karena dianggap sering mengganggunya. Pekerjaan sehari-hari Squidward adalah menjadi kasir di rumah makan Krusty Krab. Tergolong pemalas. Suka bermain klarinet walaupun musik yang dilagukannya jelek. Ia sendiri menganggap dirinya seorang seniman hebat dan orang yang pintar. Memiliki seorang saingan sejak SMA bernama Squilliam Fancyson.

Gambar 2.3 Squidward Tentacles

d. Mr. Krabs

Eugene H. Krabs adalah seekor kepiting pemilik restoran Krusty Krab yang terobsesi dengan uang. Karena karena ia terlalu mencintai uang, ia sampai-sampai tidak rela memberi uang satu sen pun. Mr. Krabs mempunyai pesaing yang bernama Sheldon Plankton, pemilik restoran Chum Bucket yang daganganya tidak pernah laku. Gambaran umum Mr. Krab adalah kepiting yang tamak dan pelit dan Spongebob seringkali menjadi korban ketamakannya. Mr. Krabs memiliki putri bernama Pearl, dan ibu Mr. Krab sudah tua.

Gambar 2.3 **Mr. Krabs**

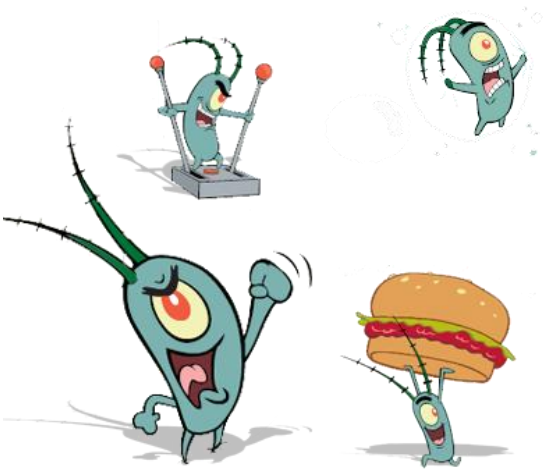
e. Sandy Cheeks

Sandy Cheeks (Sandy si Tupai) - adalah seekor tupai atau bajing yang tinggal di dalam laut. Karena Sandy merupakan binatang darat, kesehariannya di Bikini Bottom mengharuskannya memakai baju astronot yang berisi udara pada helmnya sehingga ia bisa bernapas di dalam laut. Sandy menyukai karate, dan senang aksi-aksi berbahaya. Sandy sendiri berasal dari Texas, namun kini bertempat tinggal di Bikini Bottom dan menjadi sahabat Spongebob. Tempat tinggalnya adalah sebuah kubah akuarium kedap air dan berisi udara. Ciri khas rumahnya adalah mempunyai pohon besar sebagai tempat tinggalnya. Spongebob dan Patrick bila berkunjung harus terus menerus memakai mangkuk penuh berisi air yang dipakai terbalik menutupi kepala mereka. Sandy sendiri bekerja sebagai ilmuwan di Treedome Corporation dengan bosnya yang monyet. Di episode Chimps Ahoy, dia hampir meninggalkan Bikini Bottom karena percobaan yang dia buat dianggapnya selalu salah

Gambar 2.3 Sandy Cheeks

f. Sheldon Plankton

Sheldon J. Plankton - pemilik rumah makan Chum Bucket yang sepi pengunjung. Plankton percaya bahwa bila ia berhasil mencuri resep Kraby Patty maka rumah makannya akan laku keras. Jadi segala daya upaya ia kerahkan untuk mencuri resep dari rumah makan saingannya ini, namun upayanya ini tidak pernah berhasil dan mengakibatkan kesialan untuk dirinya akhirnya di episode 'New Leaf' dia mengganti restorannya dengan toko souvenir. Memiliki istri berwujud sebuah komputer bernama Karen.



Gambar 2.4. Sheldon Plankton

g. Gary si Siput

Gary - adalah seekor siput peliharaan Spongebob. Berbunyi seperti kucing dan memiliki sifat yang pendiam dan baik. Ia tidak suka mandi dan hanya saat akan dimandikanlah Gary membuat onar.

Gambar 2.4. Gary si Siput

2.2. Java

Java adalah satu set produk perangkat lunak komputer beberapa spesifikasi dari Sun Microsystems (yang telah bergabung dengan Oracle Corporation), yang bersama-sama menyediakan sistem untuk mengembangkan perangkat lunak aplikasi dan penggelaran di lingkungan komputasi cross-platform. Java digunakan dalam berbagai platform komputasi dari perangkat embedded dan ponsel pada akhir rendah, ke server enterprise dan superkomputer pada ujung yang tinggi. Sementara kurang umum, applet Java yang kadang-kadang digunakan untuk menyediakan fungsi lebih baik dan aman saat browsing World Wide Web pada komputer desktop.

Menulis dalam bahasa pemrograman Java adalah cara utama untuk menghasilkan kode yang akan digunakan sebagai Java bytecode. Namun demikian, bytecode compiler yang tersedia untuk bahasa lain seperti Ada, JavaScript, Python, dan Ruby. Bahasa baru beberapa telah dirancang untuk berjalan secara native pada Java Virtual Machine (JVM), seperti Scala, Clojure dan Groovy. Sintaks Java meminjam banyak dari C dan C + +, namun fitur berorientasi objek dimodelkan setelah Smalltalk dan Objective-C [7]. Java menghilangkan tertentu tingkat rendah konstruksi seperti pointer dan memiliki model memori yang sangat sederhana di mana setiap objek yang dialokasikan pada tumpukan dan semua variabel jenis objek referensi. Manajemen memori ditangani melalui pengumpulan sampah terpadu otomatis dilakukan oleh JVM.

Pada tanggal 13 November 2006, Sun Microsystems membuat sebagian besar implementasi dari Java tersedia di bawah GNU General Public

2.3 Platform

Sebuah edisi dari platform Java adalah nama untuk bundel program terkait dari Sun yang memungkinkan untuk mengembangkan dan menjalankan program yang ditulis dalam bahasa pemrograman Java. Platform ini tidak spesifik untuk setiap prosesor satu atau sistem operasi, melainkan mesin eksekusi (disebut mesin virtual) dan compiler dengan satu set library yang diimplementasikan untuk berbagai hardware dan sistem operasi sehingga program Java dapat berjalan identik pada semua dari mereka.

Java Card: Sebuah teknologi yang memungkinkan kecil aplikasi berbasis Java (applet) yang akan dijalankan dengan aman pada smart card dan mirip

kecil-memori perangkat. Java ME (Micro Edition): Menentukan set yang berbeda dari perpustakaan (dikenal sebagai profil) untuk perangkat dengan penyimpanan terbatas, layar, dan kapasitas daya. Sering digunakan untuk mengembangkan aplikasi untuk perangkat mobile, PDA, set-top box TV, dan printer.

Java SE (Standard Edition): Untuk tujuan umum digunakan pada PC desktop, server, dan perangkat sejenis.

Java EE (Enterprise Edition): Java SE ditambah berbagai API yang berguna untuk multi-tier client-server aplikasi perusahaan. Platform Java terdiri dari beberapa program, masing-masing menyediakan sebagian dari kemampuan secara keseluruhan. Sebagai contoh, compiler Java, yang mengubah kode sumber Java menjadi Java bytecode (bahasa perantara untuk JVM), disediakan sebagai bagian dari Java Development Kit (JDK). Java Runtime Environment (JRE), melengkapi JVM dengan compiler just-in-time (JIT), mengubah bytecode menengah ke kode mesin asli on the fly. Serangkaian luas perpustakaan juga merupakan bagian dari platform Java.

Komponen penting dalam platform adalah bahasa Jawa compiler, perpustakaan, dan lingkungan runtime di mana Java bytecode peralihan "mengeksekusi" menurut aturan yang ditetapkan dalam spesifikasi mesin virtual.

2.4. Netbeans

NetBeans adalah sebuah lingkungan pengembangan terpadu (IDE) untuk mengembangkan terutama dengan Java, tetapi juga dengan bahasa lain, khususnya di PHP, C / C + +, dan HTML5. Ini juga merupakan platform aplikasi framework untuk aplikasi Java desktop dan lain-lain. NetBeans IDE ditulis dalam Java dan dapat berjalan di Windows, OS X, Linux, Solaris dan platform

lain yang mendukung JVM kompatibel. Platform NetBeans memungkinkan aplikasi untuk dikembangkan dari satu set modular komponen software yang disebut modul. Aplikasi berbasis platform NetBeans (termasuk IDE NetBeans itu sendiri) dapat diperpanjang oleh pengembang pihak ketiga

2.4.1 Platform NetBeans

Platform NetBeans adalah framework yang dapat digunakan kembali untuk menyederhanakan pengembangan aplikasi desktop Java Swing. Bundel NetBeans IDE untuk Java SE berisi apa yang dibutuhkan untuk memulai mengembangkan plugin NetBeans dan aplikasi berbasis NetBeans Platform, tidak ada SDK tambahan diperlukan.

Aplikasi dapat menginstal modul secara dinamis. Setiap aplikasi dapat mencakup modul Update Center untuk mengijinkan pengguna aplikasi untuk men-download upgrade digital ditandatangani dan fitur baru secara langsung ke dalam aplikasi yang berjalan. Menginstal ulang upgrade atau rilis baru tidak memaksa pengguna untuk men-download keseluruhan aplikasi lagi. Platform ini menawarkan layanan dapat digunakan kembali umum untuk aplikasi desktop, memungkinkan pengembang untuk fokus pada logika yang spesifik terhadap aplikasi mereka. Di antara fitur-fitur dari platform adalah: User interface manajemen (misalnya menu dan toolbar) Pengguna pengaturan manajemen

Manajemen penyimpanan (tabungan dan pemuatan setiap jenis data)jendela manajemen Wizard framework (mendukung langkah-demi-langkah dialog)NetBeans Visual PerpustakaanTerpadu pengembangan alat NetBeans

IDE adalah gratis, open source, cross-platform IDE dengan built-dukungan-untuk Bahasa Pemrograman Java

BAB III

PEMBAHASAN DAN PERANCANGAN

3.1 Pembahasan

Perancangan perangkat lunak permainan Sponge fight pada ini melalui beberapa tahapan proses yaitu :

1. Perancangan Gambar Spongebob .

   Spongebob yang digunakan dalam perangkat lunak ini terdiri dari tujuh buah bentuk kotak yang dikombinasikan.

   a. Spogebob bermotif kotak dengan jumlah satu sampai dengan enam buah titik motif kotak

   b. Spongebob tanpa motif atau bermotif polos

2. .Perancangan Animasi dan Suara.

   Effek animasi dan suara dari sponge fight dicari pada sumber-sumber di *internet*, kemudian disesuaikan dengan kebutuhan dari system .

3. Perancangan Permainan.

   Permaianan yang akan dibuat diracang sesuai dengan permainan spongebob sebenaranya, yakni pemain hanya dijikan untuk menggerakkan spongebob yang sedang dimainkan.

4. Penentuan Pemenang.

   Pemain dikatakan menang dalam pertandingan, jika kesehatan pemain yang dimiliki telah habis terlebih dahulu dari pada pemain lainnya.

5. Perhitungan *Score*.

Setiap kemenaganan dalam pertandingan, ditampilkan pada menu score..

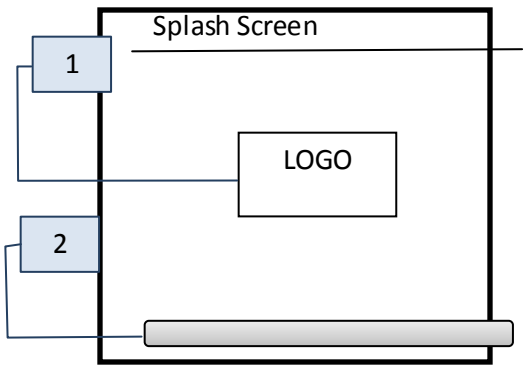
3.2 Perancangan

Perangkat lunak game strategi spongebob fight dirancang dengan menggunakan bahasa pemrograman *java* yang di buat di netbeans.

Perangkat lunak permainan ini beberapa tampilan yaitu :

1. Tampilan *Splash Screen*.
2. Tampilan Menu.
3. Tampilan Player.
4. Tampilan Score.
5. Tampilan Permainan.

3.2.1 Tampilan Splash Screen



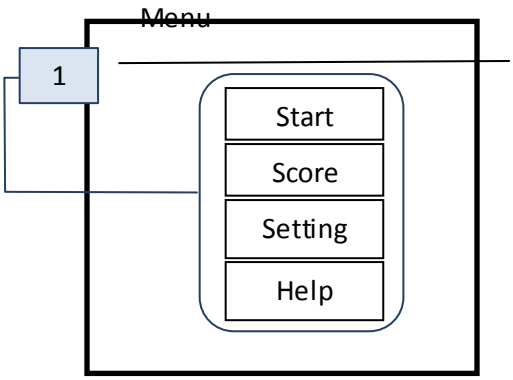
Gambar 3.1 Rancangan Tampilan *Splash Screen*

Keterangan :

1 : logo pendukung aplikasi.

2 : progress bar waktu tunggu untuk halaman splash screen.

3.2.2 Tampilan Menu



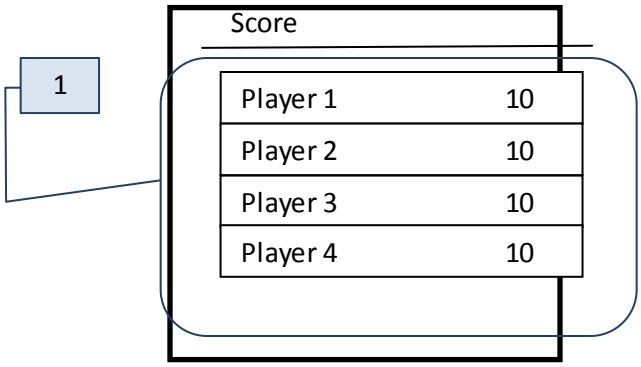
Gambar 3.2 Rancangan Tampilan *Menu*

Keterangan :

1 : tombol perangkat lunak:

a. start untuk memulai permainan
b. score untuk menampilkan nilai dan nama pemain dari permainan yang sudah dimainkan
c. setting untuk menampilkan halaman pengaturan suara
d. help menampilkan halaman bantuan.

3.2.3 Tampilan Score

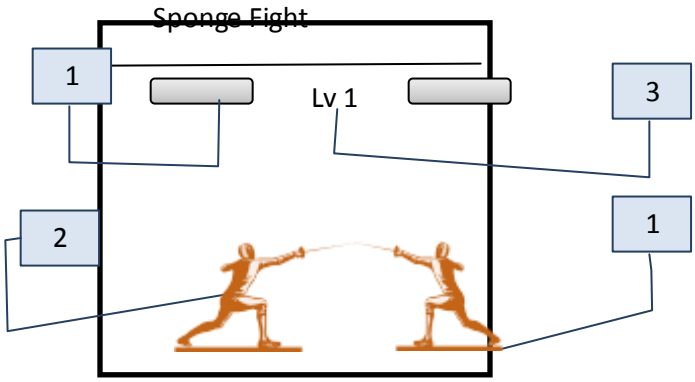


Gambar 3.3 Rancangan Tampilan *Score*

Keterangan :

1 : daftar nilai dan nama pemain yang telah bermain, dimana nilai tertinggi berada pada urutan paling atas.

3.2.4 Tampilan Permaian



Gambar 3.3 Rancangan Tampilan *Permainan*

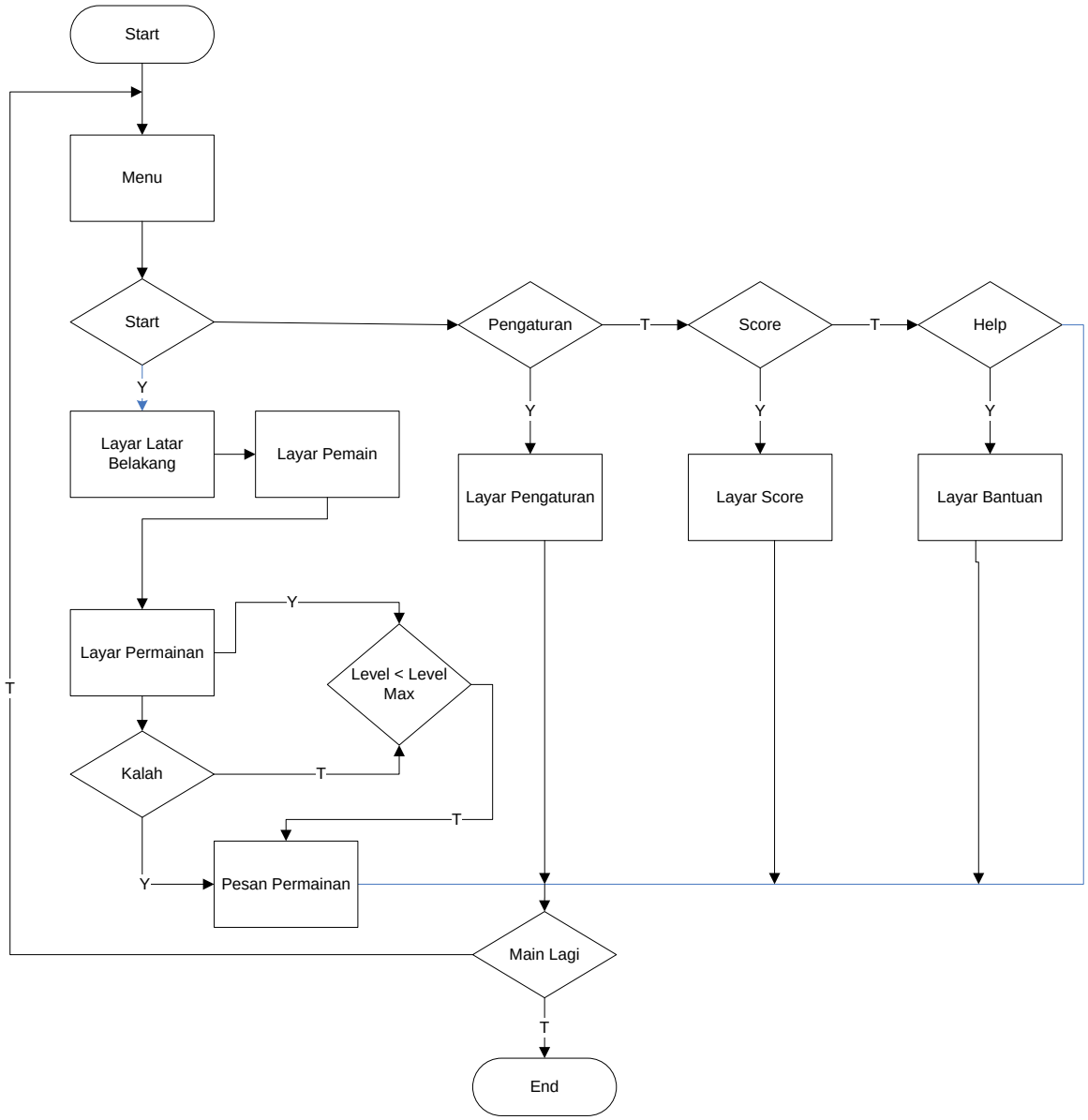
Keterangan :

1 : status kesehatan pemain yang sedang dimainkan

2 : posisi item pemain:

3 : status level yang dimainkan

3.2.5 *Flowchart* Sistem



Keterangan :

1 .Pertama-tama setelah proses halaman splash, kemudian akan muncul halaman menu.

2. Pada menu utama adalah start, pada menu ini akan ditampilkan halaman permainan, selama proses permainan berlangsung dan pemain belum kalah maka ditampilkan halaman permainan.

3. Jika pada waktu permainan kalah, maka akan ditampilkan halaman pesan permainan.
4. Pada menu setting , akan ditampilkan layar pengaturan suara
5. Pada menu score, akan ditampilkan halaman yang berisikan daftar nilai dan nama pemain pada permainan yang telah dimainkan.
6. Pada menu help, akan ditampilkan halaman bantuan
7. Jika pemain ingin melanjutkan permainan lagi, maka akan ditampilakan halaman menu

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1. Implementasi Sistem

4.1 Konfigurasi Game ”Sponge Fight”

Untuk dapat menjalankan aplikasi *game* ini dengan baik, maka diperlukan perangkat keras dan perangkat lunak yang memadai. Adapun spesifikasi perangkat keras dan perangkat lunak yang diperlukan antara lain :

1. Komputer Pentium IV dengan kecepatan minimal 2,20 GHz.
2. RAM minimal 2 Gb.
3. Ruang hardisk minimal 7,74 Mb.
4. Monitor VGA / SWGA.
5. Sistem operasi minimal Windows Seven 7 Ultimate.

4.1.1. Tampilan *Splash screen*

Hasil pengujian dari *Game Spongebob Fighter* pada tugas akhir ini dengan tampilan awal adalah *Splash Screen game* . *Splash Screen* akan keluar dalam kurun waktu beberapa detik, kemudian tampilan akan langsung masuk pada menu utama. Gambar *Splash Screen* dapat dilihat pada gambar dibawah ini :

Gambar 4.2 Potongan Program *Splash Screen*

Potongan program diatas digunakan untuk membuat *Splash Screen* pada layar awal *game*.

4.1.2. Tampilan Menu Utama

Pada Menu Utama ini terdapat 4 sub menu yaitu *Start*, *Score*, *Setting* dan *Help* Berikut ini adalah gambar menu utama tersebut :

Gambar 4.4 Potongan Program Menu Utama

Potongan program diatas merupakan potongan program utama yang digunakan untuk memanggil *class-class* pada *game*, seperti *class Splash* untuk menampilkan *Splash Screen*, *class Display* untuk menampilkan tampilan *game*, *class Games* untuk menampilkan papan games, *class Setting* untuk menampilkan pengaturan *game*, *class scoreActiviy* digunakan untuk menampilkan tampilan *score*, *Method startApp* merupakan *method* yang akan dijalankan pada saat aplikasi *dieksekusi*. Dan *method* tampilMenu diatas merupakan *method* yang digunakan untuk membuat tampilan menu dan ikon untuk masing-masing menu.

4.1.3 Tampilan Start

Pada menu ini terdapat 2 menu yaitu *Backgraund* untuk memilih suatu tempat untuk permainan. Berikut ini adalah gambar menu tersebut :

Gambar 4.1.3 Menu *Start*

4.1.4. Tampilan *Score*

Pada menu ini digunakan untuk menampilkan daftar dari pemain dan nilai dari permainan yang telah dimainkan, dimana pemain yang memiliki nilai paling tinggi, memiliki urutan paling atas. Berikut ini adalah tampilan menu *score* :

Gambar 4.1.4 Menu Score

4.1.5. Tampilan Menu *Setting*

Pada menu ini akan ditampilkan pilihan pengaturan suara yaitu Enable Sound dan disable Sound.. Berikut ini adalah gambar menu tersebut :

Gambar 4.1.5 Sound Setting

Keterangan :

- Enable Sound : Mengaktifkan Suara
- Disable Sound : Menonaktifkan Suara

4.1.6. Tampilan Menu *Help*

Pada menu ini akan ditampilkan pilihan pengaturan keybord controls untuk menjalankan karakter pada permainan game sponge fight

Gambar 4.1.6 Menu Help

4.2. Tampilan Level Game level 0

Berikut adalah beberapa contoh tampilan *level game "Sponge Fight"*. Pada setiap *level* akan tampak meteran nyawa *player* pada pojok kiri atas dan di sebelah kanannya terdapat panel penunjuk *level*.pada level ini hanya memiliki tingkat kesulitan sedang

Gambar Tampilan Level 0

4.3 level 1

Dilevel satu ini memiliki tingkat kesulitan mengalahkan lawan lebih susah karena tingkat kesehatan lawan meningkat jika di serang dan jika menang akan ke level berikutnya

Gambar Tampilan Level 1

4.4 level 2

Di level 2 tingkat kesulitan dalam mengalahkan memiliki tingkat kesulitan mengalahkan lawan lebih rumit karena tingkat kesehatan lawan meningkat jika di serang dan jika menang akan muncul ”Player 1 Win” dan akan muncul tombol ” Press any Continue klik ’OK” ke level berikutnya

Gambar Tampilan Level 2

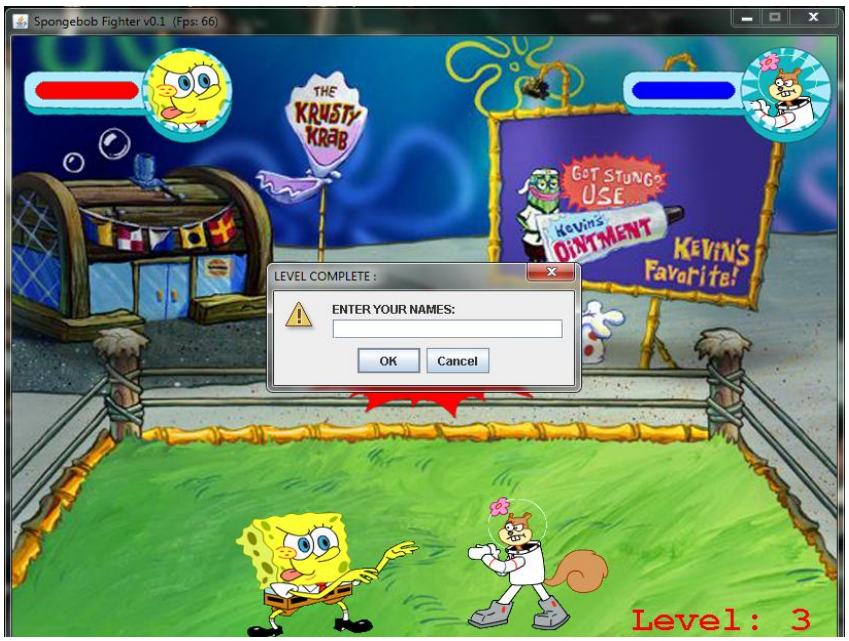
4.5 level 3

Di level 3 tingkat kesulitan dalam mengalahkan memiliki tingkat kesulitan mengalahkan lawan lebih rumit karena tingkat kesehatan lawan meningkat jika di serang dan kita bisa berkurang kesehatan pemain. Jika kita menang akan mendapat nilai karena ini level terahir dalam permainan ini.

Gambar Tampilan Level 3

4.6 Level Complete

Di level inilah sponge fight menghadapi musuh berat dari chapter 3 dan mengalahkannya untuk memenangkan game ini dan isi nama anda di dalam kolom



Gambar Tampilan Level 3

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah menyelesaikan perancangan perangkat lunak permainan Spengebob pada Java yang dapat dimainkan, penulis menarik kesimpulan sebagai berikut :

1. Game dimainkan hanya untuk 2 pemain (komputer dan user).
2. Tingkat kesulitan permainan game ini maksimal sampai level 3.

5.2 Saran

Penulis memberikan beberapa saran sebagai berikut :

1. Game dimainkan secara online. Dengan online bisa dimainkan lebih dari 2 orang pemain
2. Penambahan karakter dan tingkat kesulitan fighter.
3. Pilihan Seting tempat bermain di tambahkan

DAFTAR PUSTAKA

[1]. Campione, Mary, Walrath, Katy & Huml, Alison. 2000. The JavaT^{M} Tutorial, Third Edition : A Short Course on The Basic, Addiosion Wessley

[2]. Darwin S.2003.Java Cookbook,O'Reilly

[3]. Exckel, Bruce.2000.Thinking in Java,2nd Edition, Release 11, Prentice-Hall

[4]. Mullholland, Andrew & Murphy, Glen.2003.Java 1.4 Game Programming, Wardrobe

[5].http://id.wikipedia.org/wiki/Daftar_karakter_SpongeBob_SquarePants

akses pada tanggal : 10 January 2013

[6]. http://en.wikipedia.org/wiki/Java_%28software_platform%29

akses pada tanggal : 10 Januari 2013

[7]. http://netbeans.org/features/platform/

akses pada tanggal : 10 Januari 2013

LAMPIRAN

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

NAMA	: ACHMAD TAUFIQ
NIM	: 07.12.546
JURUSAN	: Teknik Elektro S-1
KONSENTRASI	: Teknik Komputer dan Informatika
MASA BIMBINGAN	: Semester Genap Tahun Akademik 2012-2013
JUDUL	: **RANCANG BANGUN GAME STRATEGI SPONGEBOB DENGAN BAHASA PEMROGRAMAN JAVA SCRIPT MENGGUNAKAN NETBEAN IDE 6.0**

Dipertahankan dihadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1) pada :

Hari	: Senin
Tanggal	: 18 Februari 2013
Dengan Nilai	: 74,55 B+

PANITIA UJIAN SKRIPSI

Ketua Majelis Penguji	Sekretaris Majelis Penguji
M.Ibrahim Ashari ST, MT	**Dr. Eng. Aryuanto Soetedjo, ST, MT**
NIP.P.1030100358	**NIP.P.1030800417**

ANGGOTA PENGUJI

Dosen Penguji I	Dosen Penguji II
Dr. Eng. Aryuanto Soetedjo, ST, MT	**Bambang Prio Hartono, ST,MT**
NIP.P.1030800417	**NIP.Y.1028400082**

CANVAS GAME

```
public class CanvasGame{


//  static void doThis(){
//     JFrame frame=new JFrame("Spongebob Fighter v0.1");
//     JCanvas canvas=new  JCanvas();
//     BufferedImage backImage = canvas.loadImage("src\\data\\back.pgs");
//     BufferedImage p1LifeImage = canvas.loadImage("src\\data\\status.pgs");
//     BufferedImage p1IconImage = canvas.loadImage("src\\data\\s1.pgs");
//     BufferedImage p2IconImage = canvas.loadImage("src\\data\\s2.pgs");
//
////     PlayerSprite player1=new PlayerSprite(canvas, "src\\data\\1_1.pgs","1");
////     PlayerSprite player2=new PlayerSprite(canvas, "src\\data\\2_1.pgs","2");
//
//
//     frame.setSize(backImage.getWidth(),backImage.getHeight());
//     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
//     frame.setAlwaysOnTop(true);
//     frame.setResizable(false);
//     java.awt.Dimension screenSize =
java.awt.Toolkit.getDefaultToolkit().getScreenSize();
//     java.awt.Dimension dialogSize = frame.getSize();
//     frame.setLocation((screenSize.width-dialogSize.width)/2,(screenSize.height-
dialogSize.height)/2);
//
//     frame.add(canvas);
//     frame.setVisible(true);
```

```
//
//    JEventQueue events=new JEventQueue();
//    events.listenTo(canvas,"canvas");
//
//
//
//    while(true){
//        EventObject event=events.waitEvent();
//        if(events.isKeyReleased(event)){
//            switch (events.getKeyCode(event) ){
//                case KeyEvent.VK_LEFT:
//                    System.out.println("Left");
//                    break;
//                case KeyEvent.VK_RIGHT:
//                    System.out.println("Right");
//                    break;
//                case KeyEvent.VK_UP:
//                    System.out.println("Up");
//                    break;
//                case KeyEvent.VK_DOWN:
//                    System.out.println("Down");
//                    break;
//
//
//            }
//        }
//        canvas.startBuffer();
```

```
//          canvas.clear();
//          canvas.drawImage(backImage, 0, 0);
//          canvas.drawImage(p1LifeImage,10, 10);
//          canvas.drawImage(p1LifeImage, frame.getWidth()-p1LifeImage.getWidth()-
10, 10);
//          canvas.setColor(Color.RED);
//          canvas.fillRoundRect( 22, 46, 100, 18, 20, 20);
//          canvas.setColor(Color.GRAY);
//          canvas.drawRoundRect(  22, 46, 100, 18, 20, 20);
//
//          canvas.drawImage(p1IconImage,131, 15);
//          canvas.drawImage(p2IconImage,frame.getWidth()-p1LifeImage.getWidth()
+115, 15);
//
//          canvas.setColor(Color.BLUE);
//          canvas.fillRoundRect(frame.getWidth()-p1LifeImage.getWidth() , 46, 100, 18,
20, 20);
//          canvas.setColor(Color.GRAY);
//          canvas.drawRoundRect(frame.getWidth()-p1LifeImage.getWidth() , 46, 100,
18, 20, 20);
//
////          canvas.drawImage(player1.getPlayers(),
////                  frame.getHeight() - (frame.getWidth() / 5),frame.getHeight() -
player1.getHeight() - 40);
////          canvas.drawImage(player2.getPlayers(),
////                  (frame.getWidth() / 5),frame.getHeight() - player2.getHeight() - 40);
//
//          canvas.endBuffer();
//
```

```
//     }
//  }
}
```

ENTITY

```
import java.awt.Graphics;
import java.awt.Rectangle;

/**
 * An entity represents any element that appears in the game. The
 * entity is responsible for resolving collisions and movement
 * based on a set of properties defined either by subclass or externally.
 *
 * Note that doubles are used for positions. This may seem strange
 * given that pixels locations are integers. However, using double means
 * that an entity can move a partial pixel. It doesn't of course mean that
 * they will be display half way through a pixel but allows us not lose
 * accuracy as we move.
 *
 * @author Kevin Glass
 */
public abstract class Entity {
        /** The current x location of this entity */
        protected double x;
        /** The current y location of this entity */
```

```
    protected double y;
    /** The sprite that represents this entity */
    protected Sprite sprite;
    /** The current speed of this entity horizontally (pixels/sec) */
    protected double dx;
    /** The current speed of this entity vertically (pixels/sec) */
    protected double dy;
    /** The rectangle used for this entity during collisions  resolution */
    private Rectangle me = new Rectangle();
    /** The rectangle used for other entities during collision resolution */
    private Rectangle him = new Rectangle();
protected  int Blood;
protected States states;
protected boolean isHitting = false;
protected int idxMoved;
protected String pID;
protected PlayWave playWaveKick ;
protected PlayWave playWaveJap;
protected PlayWave playWaveLong;


public void setBlood(int Blood) {
  if (Blood <= 100) this.Blood = Blood;
}

public int getBlood() {
    return Blood;
```

```
}
public void setStates (States states) {
        this.states = states;
}

public String getpID() {
    return pID;
}

public States getStates() {
        return states;
}

 public boolean isIsHitting() {
    return isHitting;
}

    /**
     * Construct a entity based on a sprite image and a location.
     *
     * @param ref The reference to the image to be displayed for this entity
     * @param x The initial x location of this entity
     * @param y The initial y location of this entity
     */
    public Entity(String ref,int x,int y) {
                this.sprite = SpriteStore.get().getSprite(ref);
                this.x = x;
```

```
            this.y = y;

    }

    /**
     * Request that this entity move itself based on a certain ammount
     * of time passing.
     *
     * @param delta The ammount of time that has passed in milliseconds
     */
    public void move(long delta) {
            // update the location of the entity based on move speeds
            x += (delta * dx) / 1000;
            y += (delta * dy) / 1000;
    }

    /**
     * Set the horizontal speed of this entity
     *
     * @param dx The horizontal speed of this entity (pixels/sec)
     */
    public void setHorizontalMovement(double dx) {
            this.dx = dx;
    }

public void setX(double x) {
   if (x>10 && x+230 <=800) this.x = x;
```

```
}

    /**
     * Set the vertical speed of this entity
     *
     * @param dx The vertical speed of this entity (pixels/sec)
     */
    public void setVerticalMovement(double dy) {
            this.dy = dy;
    }
```

```
/**
 * Get the horizontal speed of this entity
 *
 * @return The horizontal speed of this entity (pixels/sec)
 */
public double getHorizontalMovement() {
	return dx;
}

/**
 * Get the vertical speed of this entity
 *
 * @return The vertical speed of this entity (pixels/sec)
 */
public double getVerticalMovement() {
	return dy;
}

/**
 * Draw this entity to the graphics context provided
 *
 * @param g The graphics context on which to draw
 */
public void draw(Graphics g) {
```

```
            sprite.draw(g,(int) x,(int) y);
    }

    /**
     * Do the logic associated with this entity. This method
     * will be called periodically based on game events
     */
    public void doLogic() {
    }

    /**
     * Get the x location of this entity
     *
     * @return The x location of this entity
     */
    public int getX() {
            return (int) x;
    }

    /**
     * Get the y location of this entity
     *
     * @return The y location of this entity
     */
    public int getY() {
```

```
                return (int) y;
            }

            /**
             * Check if this entity collised with another.
             *
             * @param other The other entity to check collision against
             * @return True if the entities collide with each other
             */
            public boolean collidesWith(Entity other) {
            int devider = 4;
                me.setBounds((int) x + (sprite.getWidth()/devider),(int)
y,sprite.getWidth() - (sprite.getWidth()/devider),sprite.getHeight());
                him.setBounds((int) other.x+ (other.sprite.getWidth()/devider)
,(int) other.y,other.sprite.getWidth() -
(other.sprite.getWidth()/devider),other.sprite.getHeight());

                return me.intersects(him);
            }
            public void doanimate(long delta){

        }

            /**
             * Notification that this entity collided with another.
             *
             * @param other The entity with which this entity collided.
```

```
 */
        public abstract void collidedWith(Entity other);
}
```

FRM GAME

```
import java.awt.Canvas;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.awt.image.BufferStrategy;
import java.util.ArrayList;
import java.util.Random;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;

/*
 * To change this template, choose Tools | Templates
```

```
 * and open the template in the editor.
 */

/**
 *
 * @author Mbah Marijan Okek
 */
public class FrmGame extends Canvas {



    private double moveSpeed = 300;
    private long lastFire = 0;
   private long firingInterval = 500;

   private BufferStrategy strategy;

    private boolean gameRunning = true;
   private ArrayList entities = new ArrayList();
   private ArrayList removeList = new ArrayList();



    private boolean waitingForKeyPress = true;
    private boolean leftPressed = false;
    private boolean rightPressed = false;
    private boolean japPressed = false;
```

```
private boolean kickPressed = false;
private boolean longPressed = false;



    private boolean logicRequiredThisLoop = false;
    private long lastFpsTime;
    private int fps;
    private String windowTitle = "Spongebob Fighter v0.1";
    private JFrame container;

private Image imgBack,imgBack2,imgStatus,imgSplash,imgMenu;
private Image imgStatusP1,imgStatusP2;
private Image imgBtnUP,imgBtnDown;
private Image imgP1,imgP2;
private Image imgIconP1,imgIconP2;
private Image imgAlert;

public  Entity Player2;
public  Entity Player1;

public boolean isHit =false;
private Font largeBannerFont;
private Font smallBannerFont;
private boolean isEnd=false;
```

```
        private GameStates GAME_STATES;
        FrmHelp frmHelp;
        FrmSetting frmSetting;
        FrmScores frmScores;
        public int Levels =0;
        public int scores =0;
        static PlayWave playWave;
        private String[] strMSG={"Aw.!","Huh.!!","Yuo.."};
        private PlayWave playWaveLoss = new PlayWave();
        private PlayWave playWaveWin = new PlayWave();
        private PlayWave playJapWave;
        private PlayWave playLongWave;
        private PlayWave playKickWave;


           public FrmGame() {
               imgBack = new
ImageIcon(getClass().getResource("/data/back.pgs")).getImage();
               imgBack2 = new
ImageIcon(getClass().getResource("/data/back2.pgs")).getImage();
               imgMenu = new
ImageIcon(getClass().getResource("/data/menu.pgs")).getImage();
               imgStatus = new
ImageIcon(getClass().getResource("/data/status.pgs")).getImage();
               imgSplash = new
ImageIcon(getClass().getResource("/data/splash.pgs")).getImage();
               imgStatusP1 = new
ImageIcon(getClass().getResource("/data/s1.pgs")).getImage();
```

```
                imgStatusP2 = new
ImageIcon(getClass().getResource("/data/s2.pgs")).getImage();

                imgP1 = new
ImageIcon(getClass().getResource("/data/bicon.pgs")).getImage();

                imgP2 = new
ImageIcon(getClass().getResource("/data/bicon2.pgs")).getImage();


                imgIconP1 = new
ImageIcon(getClass().getResource("/data/s1icon.png")).getImage();

                imgIconP2 = new
ImageIcon(getClass().getResource("/data/s2icon.png")).getImage();


                imgBtnUP = new
ImageIcon(getClass().getResource("/data/btn_up.png")).getImage();

                imgBtnDown = new
ImageIcon(getClass().getResource("/data/btn_down.png")).getImage();

                imgAlert = new
ImageIcon(getClass().getResource("/data/alert.pgs")).getImage();


                frmHelp = new FrmHelp();

                frmSetting = new FrmSetting();

                frmScores = new FrmScores();

                playWave = new PlayWave();

                playJapWave = new PlayWave();

                playLongWave = new PlayWave();

                playKickWave = new PlayWave()


        ;
```

```
                playWaveLoss.setFilename("data/loss.wav");
                playWaveWin.setFilename("data/win.wav");

                GAME_STATES = GameStates.SPLASH;
                container = new JFrame("Spongebob Fighter v0.1");
                container.setSize(imgBack.getWidth(this),imgBack.getHeight(this));

container.setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_ON_CLOSE);
                container.setAlwaysOnTop(true);
                container.setResizable(false);
                java.awt.Dimension screenSize =
java.awt.Toolkit.getDefaultToolkit().getScreenSize();
                java.awt.Dimension dialogSize = container.getSize();
                container.setLocation((screenSize.width-
dialogSize.width)/2,(screenSize.height-dialogSize.height)/2);

                    JPanel panel = (JPanel) container.getContentPane();
                    panel.setPreferredSize(new
Dimension(container.getWidth(),container.getHeight()));
                    panel.setLayout(null);

                    setBounds(0,0,container.getWidth(),container.getHeight());
                    panel.add(this);

                    setIgnoreRepaint(true);
```

```
                container.pack();
                container.setResizable(false);
                container.setVisible(true);



                container.addWindowListener(new WindowAdapter() {
                        public void windowClosing(WindowEvent e) {
                    isPause =true;
                  if (GAME_STATES==GameStates.MENU){
                                if
(JOptionPane.showConfirmDialog(container,"Are you sure to end the game?","End
Game?",JOptionPane.YES_NO_OPTION,JOptionPane.QUESTION_MESSAGE,
null)==JOptionPane.YES_OPTION){
                                    System.exit(0);
                                }
                                }else{
                                     GAME_STATES =GameStates.MENU;
                                }
                                    }
                });
            font = new Font("monospaced", Font.BOLD, 68);
                addKeyListener((KeyListener) new KeyInputHandler());
                requestFocus();
                createBufferStrategy(2);
                strategy = getBufferStrategy();
```

```
            initEntities();

    }


    private void startGame() {

       entities.clear();
            initEntities();
       timeStart =300;
            leftPressed = false;
            rightPressed = false;
            japPressed = false;
       longPressed =false;
       kickPressed =false;
       isEnd = false;
       isHit = false;
       isPause = false;


    }


    private void initEntities() {

    if (playerSelect==0){
```

```
                Player2 = new Player1Entity(this ,"data/1_1.pgs",
container.getWidth() - (container.getWidth() / 2), container.getHeight( ) -
imgP2.getHeight(this) - 40);

                    Player1 = new Player2Entity(this ,"data/2_1.pgs",
(container.getWidth() / 4), container.getHeight( ) - imgP2.getHeight(this) - 40);

            }else

            {

                Player2 = new Player1Entity(this ,"data/2_1.pgs",
(container.getWidth() / 4), container.getHeight( ) - imgP2.getHeight(this) - 40);

                    Player1 = new Player2Entity(this ,"data/1_1.pgs",
container.getWidth() - (container.getWidth() / 2), container.getHeight( ) -
imgP2.getHeight(this) - 40);

            }

            System.out.println("Player States:" +playerSelect);

            entities.add(Player1);

            entities.add(Player2);



            }



            public void updateLogic() {

                    logicRequiredThisLoop = true;

            }



            public void notifyDeath(String pID,int Minus) {
```

```
                if  (Player1.getStates()==States.HURT &&
Player2.getStates()==States.HURT){
                    Player1.setStates(States.FREE);
                    Player2.setStates(States.FREE);
                }

                Player2.setX(Player2.getX()+ (Player1.getpID().equals("1") ? - 40 : +
40) );
                Player1.setX(Player1.getX()+ (Player1.getpID().equals("1") ? + 40 : -
40) );

                leftPressed = false;
                    rightPressed = false;
                    japPressed = false;
                longPressed =false;
                kickPressed =false;
            }

            public void notifyWin() {

                    waitingForKeyPress = true;
            }
```

```
    public void tryToFire() {

            if (System.currentTimeMillis() - lastFire < firingInterval) {
                return;
            }
            lastFire = System.currentTimeMillis();

    }
private Font font ;
 public void drawTeks(String s,int x, int y, Graphics2D g2d){

    g2d.setColor(Color.BLACK );
    g2d.drawString(s, x, y);
    g2d.setColor(Color.ORANGE);
    g2d.drawString(s, x+1, y+1);
}

private static  Random randStates= new Random();
int random(int size)
{
    return (randStates.nextInt() & 0x7FFFFFFF) % size + 1;
}
private boolean isPause=false;
int timeStart =300;
int backSelect =0;
```

```
int playerSelect =0;
int menuSelect =0;
boolean isAdd =false;
   public void gameLoop() {
         long lastLoopTime = System.currentTimeMillis();

    int loopTime=0;

    largeBannerFont = new Font("TimesRoman", Font.BOLD, 72);
    smallBannerFont = new Font("monospaced", Font.BOLD, 36);
    int idx =0;
         while (gameRunning) {

               long delta = System.currentTimeMillis() - lastLoopTime;
               lastLoopTime = System.currentTimeMillis();


               lastFpsTime += delta;
               fps++;


               if (lastFpsTime >= 1000) {
                     container.setTitle(windowTitle+"  (Fps:
"+fps+")");
                     lastFpsTime = 0;
                     fps = 0;
```

```
                }



                Graphics2D g = (Graphics2D)
strategy.getDrawGraphics();
                g.setColor(Color.black);
                g.fillRect(0,0,container.getWidth(),container.getHeight());

            switch (GAME_STATES){
              case SPLASH:
                g.drawImage(imgSplash, 0, 0,  this);
                g.setColor(Color.ORANGE);
                g.fillRoundRect( 0, getHeight()-12,  idx, 12, 0, 0);



                g.dispose();

                strategy.show();
                try {
                   Thread.sleep(Math.abs(   lastLoopTime + 10 -
System.currentTimeMillis()));
                } catch (InterruptedException ex) {

                }
                if (idx >= getWidth()){
                  GAME_STATES = GameStates.MENU;
```

```
                    }else{
                        idx +=2;
                    }
                    break;
                case MENU:
                    g.drawImage(imgMenu, 0, 0,  this);
                    g.setColor(Color.ORANGE);
                    g.fillRoundRect( 0, getHeight()-12,  idx, 12, 0, 0);
                    g.setColor(Color.RED);
                    g.setFont(font);
                    drawTeks("Sponge Fight", ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2) )-70 , (container.getHeight()/2) -
(imgP1.getHeight(this)/2)-97 , g);

                    if ((leftPressed) && (!rightPressed)) {
                      if (isAdd) menuSelect--;
                      isAdd =false;
                    } else if ((rightPressed) && (!leftPressed)) {
                       if (isAdd) menuSelect ++;
                       isAdd =false;
                    } else if (japPressed){
                       japPressed =false;
                       switch (menuSelect){
                          case 0:
                             GAME_STATES = GameStates.BACK;

                             Levels =0;
```

```
            break;
        case 1:
            frmScores.setVisible(true);
            break;
        case 2:
            frmSetting.setVisible( true);
            break;
        case 3:
            frmHelp.setVisible(true);

            break;

    }

     isAdd=false;
}
 if (menuSelect<0){
    menuSelect = 3 ;
}else if (menuSelect>3){
    menuSelect=0;
}



if (menuSelect==0){
```

```
                    g.drawImage(imgBtnDown, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2)-37
, this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
32 , this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
100 , this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
167 , this);

                }else if (menuSelect==1){

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2)-37
, this);

                    g.drawImage(imgBtnDown, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
32 , this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
100 , this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
167 , this);

                }else if (menuSelect==2){

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2)-37
, this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
32 , this);
```

```
                    g.drawImage(imgBtnDown, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
100 , this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
167 , this);

                }else if (menuSelect==3){

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2)-37
, this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
32 , this);

                    g.drawImage(imgBtnUP, ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
100 , this);

                    g.drawImage(imgBtnDown , ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) , (container.getHeight()/2) - (imgP1.getHeight(this)/2) +
167 , this);

                }

                g.setFont(smallBannerFont);

                g.setColor(Color.WHITE);

                g.drawString("Start", ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) + 100 -5  , (container.getHeight()/2) -
(imgP1.getHeight(this)/2) + 7  );

                g.drawString("Score", ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) + 100 -5 , (container.getHeight()/2) -
(imgP1.getHeight(this)/2) + 73  );

                g.drawString("Setting", ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) + 75  -5, (container.getHeight()/2) -
(imgP1.getHeight(this)/2) + 140  );
```

```
                    g.drawString("Help", ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2)) + 110  -5, (container.getHeight()/2) -
(imgP1.getHeight(this)/2) + 207  );


                    g.dispose();


                    strategy.show();



                    break;
                case BACK:
                    g.drawImage(imgMenu, 0, 0,  this);
                    g.setColor(Color.ORANGE);
                    g.fillRoundRect( 0, getHeight()-12,  idx, 12, 0, 0);
                    g.setColor(Color.RED);

                     if ((leftPressed) && (!rightPressed)) {
                       backSelect=0;


                    } else if ((rightPressed) && (!leftPressed)) {
                        backSelect =1;
                    } else if (japPressed){
                        japPressed =false;
                        GAME_STATES = GameStates.PLAYER;
                    }
```

```
                    g.setFont(font);

                    drawTeks("Background", ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2) )-70 , (container.getHeight()/2) -
(imgP1.getHeight(this)/2)-97 , g);

                    if (backSelect==0){

                       g.fillRoundRect( -4 +(container.getWidth()-
imgP1.getWidth(this))/3,-4 +((container.getHeight() - imgP1.getHeight(this)) /2 ),

                            imgP1.getWidth(this)+8,
imgP1.getHeight(this)+8,10,10 );

                    }else {

                       g.fillRoundRect( -4 +((container.getWidth()-
imgP2.getWidth(this))/3 +imgP1.getWidth(this) + 20)  ,-4 + ((container.getHeight() -
imgP2.getHeight(this)) /2 ),

                            imgP2.getWidth(this)+8,
imgP2.getHeight(this)+8,10,10 );

                    }


                    g.drawImage(imgP1, (container.getWidth()-
imgP1.getWidth(this))/3,((container.getHeight() - imgP1.getHeight(this)) /2 ), this);

                    g.drawImage(imgP2, (container.getWidth()-
imgP2.getWidth(this))/3 +imgP1.getWidth(this) + 20 ,((container.getHeight() -
imgP2.getHeight(this)) /2 ), this);


                    g.dispose();


                    strategy.show();



                    break;
```

```
                case PLAYER :
                    g.drawImage(imgMenu, 0, 0,  this);
                    g.setColor(Color.ORANGE);
                    g.fillRoundRect( 0, getHeight()-12,  idx, 12, 0, 0);
                    g.setColor(Color.RED);

                     if ((leftPressed) && (!rightPressed)) {
                       playerSelect=0;

                    } else if ((rightPressed) && (!leftPressed)) {
                        playerSelect =1;
                    } else if (japPressed){
                        japPressed =false;
                        GAME_STATES = GameStates.PLAY;
                    }

                    g.setFont(font);
                    drawTeks("Player", ((container.getWidth()/2)-
(imgBtnUP.getWidth(this)/2) )-70 , (container.getHeight()/2) -
(imgIconP1.getHeight(this)/2)-97 , g);
                    if (playerSelect==0){
                       g.fillRoundRect( -4 +(container.getWidth()-
imgIconP1.getWidth(this))/3,-4 +((container.getHeight() - imgIconP1.getHeight(this))
/2 ),
                            imgIconP1.getWidth(this)+8,
imgIconP1.getHeight(this)+8,10,10 );
                    }else {
```

```
                    g.fillRoundRect( -4 +((container.getWidth()-
imgIconP2.getWidth(this))/3 +imgIconP2.getWidth(this) + 20)  ,-4 +
((container.getHeight() - imgIconP2.getHeight(this)) /2 ),

                        imgIconP2.getWidth(this)+8,
imgIconP2.getHeight(this)+8,10,10 );

                    }


                    g.drawImage(imgIconP1, (container.getWidth()-
imgIconP1.getWidth(this))/3,((container.getHeight() - imgIconP1.getHeight(this)) /2 ),
this);

                    g.drawImage(imgIconP2, (container.getWidth()-
imgIconP2.getWidth(this))/3 +imgIconP2.getWidth(this) + 20 ,((container.getHeight() -
imgIconP2.getHeight(this)) /2 ), this);


                    g.dispose();


                    strategy.show();



                    break;
                case PLAY:


                    if (backSelect==0) {


                      g.drawImage(imgBack, 0, 0,  this);
                    } else if (backSelect==1) {
                      g.drawImage(imgBack2, 0, 0,  this);
                    }
```

```
                    g.drawImage(imgStatus,10, 10,this);

                    g.drawImage(imgStatus, container.getWidth()-
imgStatus.getWidth(this)- 10, 10,this);


                    g.setColor(Color.RED);

                    g.fillRoundRect( 22, 46,  Player1.getBlood(), 18, 20, 20);

                    g.setColor(Color.GRAY);

                    g.drawRoundRect(  22, 46, 100, 18, 20, 20);


                    g.drawImage(imgStatusP1,131, 15,this);

                    g.drawImage(imgStatusP2,container.getWidth()-
imgStatusP2.getWidth(this) - 15, 15,this);


                    g.setColor(Color.BLUE);

                    g.fillRoundRect(container.getWidth()-
imgStatus.getWidth(this) , 46, Player2.getBlood(), 18,  20, 20);

                    g.setColor(Color.GRAY);

                    g.drawRoundRect(container.getWidth()-
imgStatus.getWidth(this) , 46, 100, 18, 20, 20);



                    if (!waitingForKeyPress && !isPause && !isEnd )

                    {

                        for (int i=0;i<entities.size();i++) {

                            Entity entity = (Entity) entities.get(i);
```

```
                entity.doanimate(delta);
            }
        }

        if (!isEnd || !isPause)
        {
           for (int i=0;i<entities.size();i++) {
                Entity entity = (Entity) entities.get(i);

                entity.draw(g);
           }
        }

        for (int p=0;p<entities.size();p++) {
            for (int s=p+1;s<entities.size();s++) {
                Entity me = (Entity) entities.get(p);
                Entity him = (Entity) entities.get(s);

                if (me.collidesWith(him)) {
                    me.collidedWith(him);
                    him.collidedWith(me);
                }
            }
        }
```

```
                    entities.removeAll(removeList);
                    removeList.clear();



                    if (logicRequiredThisLoop) {
                        for (int i=0;i<entities.size();i++) {
                            Entity entity = (Entity) entities.get(i);
                            entity.doLogic();
                        }

                        logicRequiredThisLoop = false;
                    }




                    if (waitingForKeyPress) {
                        if (Levels>3){
                           timeStart=0;
                           g.setFont(smallBannerFont);
                           drawStrins(g, smallBannerFont,  "END LEVEL" );
                        }else {
                           g.drawImage(imgAlert, (container.getWidth()-
imgAlert.getWidth(this))/2,((container.getHeight() - imgAlert.getHeight(this)) /2 ),
this);
```

```
                                    g.setColor(Color.white);


                                    if (timeStart/100>=1){
                                       g.setFont(largeBannerFont);

g.drawString(String.valueOf(timeStart/100),(container.getWidth()-
g.getFontMetrics().stringWidth(String.valueOf(timeStart/100)))/2,((container.getHeight
() - imgAlert.getHeight(this)) /2 )+90);
                                    }else{
                                       g.setFont(smallBannerFont);
                                       drawStrins(g, smallBannerFont,  "START" );
                                    }
                                 }



                              }
                              g.setFont(smallBannerFont);
                              g.setColor(Color.RED);
                              g.drawString("Level: " +Levels, getWidth()-200,
getHeight()-10);
                              if ( (Player1.getStates()==States.HURT ||
Player2.getStates()==States.HURT) && (
                                    Player1.getStates()!=States.END ||
Player2.getStates()!=States.END) && !isPause )
                              {
                                 g.drawImage(imgAlert, (container.getWidth()-
imgAlert.getWidth(this))/2,((container.getHeight() - imgAlert.getHeight(this)) /2 ),
this);
                                 g.setColor(Color.white);
```

```
                    g.setFont(smallBannerFont);
                    if (Player1.getStates() ==States.HURT){
                       drawStrins(g, smallBannerFont,  "Awh..!!" );
                    }else if (Player2.getStates() ==States.HURT){
                       drawStrins(g, smallBannerFont,  "Oawh..!!" );
                    }else{
                       drawStrins(g, smallBannerFont,  "Oh..!!" );
                    }



                }
                if (
                   Player1.getStates()==States.END ||
Player2.getStates()==States.END)
                {


                   isEnd=true;
                   g.drawImage(imgAlert, (container.getWidth()-
imgAlert.getWidth(this))/2,((container.getHeight() - imgAlert.getHeight(this)) /2 ),
this);
                   g.setColor(Color.white);
                   if ( Player1.getStates()==States.END )
                   {


                      playWaveLoss.DoPlay();
                      drawStrins(g, smallBannerFont,  "PLAYER 1
LOOSE" );
```

```
                }else if (Player2.getStates()==States.END )  {

                    playWaveWin.DoPlay();

                    drawStrins(g, smallBannerFont,  "PLAYER 1 WIN" );

                }

                waitingForKeyPress = true;

            }
            if (isPause){
                g.drawImage(imgAlert, (container.getWidth()-
imgAlert.getWidth(this))/2,((container.getHeight() - imgAlert.getHeight(this)) /2 ),
this);
                g.setColor(Color.white);
                drawStrins(g, smallBannerFont,  "PAUSE" );

            }

            g.dispose();
            strategy.show();
```

```
                    Player1.setHorizontalMovement(0);
                    if ((Player1.getStates()!=States.START ||
Player2.getStates()!=States.START ||
                        Player1.getStates()!=States.HURT ||
Player2.getStates()!=States.HURT


                    )   && !waitingForKeyPress && !isEnd && !isPause)
{
                   if (isHit && !Player2.isIsHitting()){

                    if ((leftPressed) && (!rightPressed)) {
                        Player1.setHorizontalMovement(-moveSpeed);
                        Player1.setX(Player1.getX()-1);
                        Player1.setStates(States.MOVE);
                        System.out.println("Left Press");

                    } else if ((rightPressed) && (!leftPressed)) {
                        Player1.setHorizontalMovement(moveSpeed);
                        Player1.setX(Player1.getX()+1);
                        System.out.println("Right Press");
                        Player1.setStates(States.MOVE);
                    }

                    if (japPressed) {
                        Player1.setStates(States.JAP);
                        System.out.println("Jap Press");
                        japPressed =false;
```

```
                playJapWave.setFilename("data/jap.wav");
                playJapWave.DoPlay();



            }else if (kickPressed) {
                Player1.setStates(States.KICK);
                System.out.println("Kick Press");
                kickPressed =false;
                playKickWave.setFilename("data/kick.wav");
                playKickWave.DoPlay();


            }else if (longPressed){
                Player1.setStates(States.LONG);
                System.out.println("Long Press");
                longPressed =false;
                playKickWave.setFilename("data/long.wav");
                playKickWave.DoPlay();
            }
        }if (!isEnd &&  !Player1.isIsHitting()){
            loopTime++;
            if (loopTime >=150){
                loopTime=0;
                int newStates = random(3);
                System.out.println(newStates);
                Player2.setX(Player1.getX()+100);
                switch (newStates) {
                    case 0:
```

```
            Player2.setStates(States.JAP);

            playJapWave.setFilename("data/jap.wav");
            playJapWave.DoPlay();
            break;
        case 1:
            Player2.setStates(States.KICK);

            playKickWave.setFilename("data/long.wav");
            playKickWave.DoPlay();

            break;
        case 2:
            Player2.setStates(States.LONG);

            playLongWave.setFilename("data/long.wav");
            playLongWave.DoPlay();

            break;
        default:
            Player2.setStates(States.JAP);

            playJapWave.setFilename("data/jap.wav");
            playJapWave.DoPlay();

            break;
```

```
                    }
                }
            }

        }


        if (waitingForKeyPress && timeStart <1){

            scores += Player1.getBlood()*(Levels);
            Levels++;
            System.out.println("Scores : " + scores);
            if (Levels > 3){
                GAME_STATES =GameStates.MENU;
                //JOptionPane.showMessageDialog(container,
"CONGRATULATION, YOU
WIN","INFO",JOptionPane.INFORMATION_MESSAGE);
                String strs = JOptionPane.showInputDialog(container,
"ENTER YOUR NAMES: ", "LEVEL COMPLETE
:",JOptionPane.OK_CANCEL_OPTION);
                 if ( strs == null ? ("") != null : !strs.equals("") ){
                   if (strs== null || strs.trim().length()==0){
                      strs = "No Name";
                   }

                 }
                new Scores().setNewScore(strs+":"+scores);
                break;
            }
```

```
                    JOptionPane.showMessageDialog(container, "PRESS
ANY KEY TO CONTUNUE","INFO",JOptionPane.INFORMATION_MESSAGE);
                    doStart(1);

                    waitingForKeyPress =false;
                    Player2.setStates(States.START);
                    Player1.setStates(States.START);
                }else{
                    if (waitingForKeyPress && timeStart>0) timeStart--;

                }
                try {
                     Thread.sleep(Math.abs(   lastLoopTime + 15 -
System.currentTimeMillis()));
                } catch (InterruptedException ex) {

                }
                startSound.DoPlay();

                break;

            }



            }
        }
    void drawStrins(Graphics2D g, Font f ,String txt){
```

```
                g.setFont(f);
                g.drawString(txt,(container.getWidth()-
g.getFontMetrics().stringWidth(txt))/2,((container.getHeight() -
imgAlert.getHeight(this)) /2 )+80);
            }
            public void removeEntity(Entity entity) {
                removeList.add(entity);
            }

            private class KeyInputHandler extends KeyAdapter {

                private int pressCount = 1;

                public void keyPressed(KeyEvent e) {
                if (GAME_STATES == GameStates.MENU){

                    if (e.getKeyCode() == KeyEvent.VK_UP) {
                        leftPressed = true;
                    }
                    if (e.getKeyCode() == KeyEvent.VK_DOWN) {
                        rightPressed = true;
                    }
                if (e.getKeyCode() == KeyEvent.VK_ENTER) {
                        japPressed = true;
                    }
                return;
```

```
                }else if (GAME_STATES == GameStates.BACK ||
GAME_STATES == GameStates.PLAYER){


                        if (e.getKeyCode() == KeyEvent.VK_LEFT) {
                                leftPressed = true;
                        }
                        if (e.getKeyCode() == KeyEvent.VK_RIGHT) {
                                rightPressed = true;
                        }
                    if (e.getKeyCode() == KeyEvent.VK_ENTER) {
                                japPressed = true;
                        }
                   return;
                }
                        if (waitingForKeyPress ||
Player2.getStates()==States.END) {
                                return;
                        }
                    isHit =true;


                        if (e.getKeyCode() == KeyEvent.VK_LEFT) {
                                leftPressed = true;
                        }
                        if (e.getKeyCode() == KeyEvent.VK_RIGHT) {
                                rightPressed = true;
                        }
                        if (e.getKeyCode() == KeyEvent.VK_SPACE) {
                                japPressed = true;
```

```
            }
        if (e.getKeyCode() == KeyEvent.VK_A) {
                    longPressed = true;
            }
        if (e.getKeyCode() == KeyEvent.VK_S) {
                    kickPressed  = true;
            }
        if (e.getKeyCode() == KeyEvent.VK_P) {

            }
        isHit =true;

        }

        public void keyReleased(KeyEvent e) {
    if (GAME_STATES == GameStates.MENU){

            if (e.getKeyCode() == KeyEvent.VK_UP) {
                    leftPressed = false;
            }
            if (e.getKeyCode() == KeyEvent.VK_DOWN ) {
                    rightPressed = false;
            }
        if (e.getKeyCode() == KeyEvent.VK_ENTER) {
                    japPressed = false;
            }
        isAdd =true;
```

```
                return;
            }else if (GAME_STATES == GameStates.BACK ||
GAME_STATES == GameStates.PLAYER){

                    if (e.getKeyCode() == KeyEvent.VK_LEFT) {
                        leftPressed = false;
                    }
                    if (e.getKeyCode() == KeyEvent.VK_RIGHT) {
                        rightPressed = false;
                    }
                if (e.getKeyCode() == KeyEvent.VK_ENTER) {
                        japPressed = false;
                    }
                isAdd =true;
                return;
            }
                    if (waitingForKeyPress ||
Player2.getStates()==States.END) {
                        return;
                    }

                    if (e.getKeyCode() == KeyEvent.VK_LEFT) {
                        leftPressed = false;
                    }
                    if (e.getKeyCode() == KeyEvent.VK_RIGHT) {
                        rightPressed = false;
                    }
```

```
                if (e.getKeyCode() == KeyEvent.VK_SPACE) {
                        japPressed = false;
                }
                if (e.getKeyCode() == KeyEvent.VK_A) {
                        longPressed = false;
                }
                if (e.getKeyCode() == KeyEvent.VK_S) {
                        kickPressed  = false;
                }
                if (e.getKeyCode() == KeyEvent.VK_P) {
                         isPause  = !isPause;
                }
                 isHit =false;
                }
                public void keyTyped(KeyEvent e) {
                        if (waitingForKeyPress ||
Player2.getStates()==States.END) {
                                if (isEnd)doStart(pressCount);
                        }

                        if (e.getKeyChar() == 27) {

                    isPause =true;
                   if (GAME_STATES==GameStates.MENU){
                                if
(JOptionPane.showConfirmDialog(container,"Are you sure to end the game?","End
Game?",JOptionPane.YES_NO_OPTION,JOptionPane.QUESTION_MESSAGE,
null)==JOptionPane.YES_OPTION){
                                System.exit(0);
```

```
                    }
                }else{
                        GAME_STATES =GameStates.MENU;
                }
                    }
                }


    }
  private void doStart(int pressCount) {

    if (pressCount == 1) {
        //waitingForKeyPress = false;
        startGame();
        pressCount = 0;
    } else {
        pressCount++;
    }
}
  static PlayWave startSound =new PlayWave();
    public static void main(String argv[]) {
            FrmGame g = new FrmGame();
       startSound.setFilename("data/ready.wav") ;
       playWave.setFilename("data/splash.wav");
       playWave.DoPlay();

            g.gameLoop();
```

```
        }
}
```

FRM HELP

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/*
 * FrmHelp.java
 *
 * Created on Jan 2, 2012, 1:40:58 AM
 */

/**
 *
 * @author Mbah Marijan Okek
 */
public class FrmHelp extends javax.swing.JFrame {

    /** Creates new form FrmHelp */
    public FrmHelp() {
        initComponents();
    }
```

```
    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        panelImage1 = new PanelImage();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("Help");
        setAlwaysOnTop(true);

        jLabel1.setFont(new java.awt.Font("Comic Sans MS", 1, 48)); // NOI18N
        jLabel1.setForeground(new java.awt.Color(255, 204, 51));
        jLabel1.setText("Keyboard Controls");
```

```
            jLabel2.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N
            jLabel2.setForeground(new java.awt.Color(255, 204, 51));
            jLabel2.setText("- Move : Left, Right");

            jLabel3.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N
            jLabel3.setForeground(new java.awt.Color(255, 204, 51));
            jLabel3.setText("- Jap : Space");

            jLabel4.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N
            jLabel4.setForeground(new java.awt.Color(255, 204, 51));
            jLabel4.setText("- Kick : S");

            jLabel5.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N
            jLabel5.setForeground(new java.awt.Color(255, 204, 51));
            jLabel5.setText("- Long Kick : A");

            jLabel6.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N
            jLabel6.setForeground(new java.awt.Color(255, 204, 51));
            jLabel6.setText("- Pause : P");

            jLabel7.setFont(new java.awt.Font("Comic Sans MS", 1, 24)); // NOI18N
            jLabel7.setForeground(new java.awt.Color(255, 204, 51));
            jLabel7.setText("- Commit : Enter");

            javax.swing.GroupLayout panelImage1Layout = new
javax.swing.GroupLayout(panelImage1);
```

```
            panelImage1.setLayout(panelImage1Layout);
            panelImage1Layout.setHorizontalGroup(
panelImage1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                .addGroup(panelImage1Layout.createSequentialGroup()
                    .addGap(123, 123, 123)
.addGroup(panelImage1Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
                        .addComponent(jLabel2)
                        .addComponent(jLabel1)
                        .addComponent(jLabel3)
                        .addComponent(jLabel4)
                        .addComponent(jLabel5)
                        .addComponent(jLabel6)
                        .addComponent(jLabel7))
                    .addContainerGap(110, Short.MAX_VALUE))
            );
            panelImage1Layout.setVerticalGroup(
panelImage1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                .addGroup(panelImage1Layout.createSequentialGroup()
                    .addGap(69, 69, 69)
                    .addComponent(jLabel1)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel2)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jLabel7)
                    .addContainerGap(51, Short.MAX_VALUE))
                );


                javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
                getContentPane().setLayout(layout);
                layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(panelImage1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```