About The Team   
Kami adalah Calogerus Draconis, beranggotakan:  
Rita Sarah (13512009)  
Denny Astika Herdioso (13512011)  
Cilvia Sianora Putri (13512027)  
Willy (13512065)  
Dariel Valdano(13512079)

Background

Masalah yang akan diselesaikan adalah untuk membuat sebuah peliharaan didalam program yang akan kami buat. Dengan merawat peliharaan tersebut player dapat menghilangkan rasa kebosanan.Program ini juga dapat digunakan sebagai simulasi memelihara dragon, atau dijadikan alternatif bintang yang dapat dipelihara karena player tidak dapat memelihara binatang .  
  
Alasan kami memilih topik ini adalah untuk menyediakan sarana bagi orang-orang yang ingin memelihara dragon dan mengisi waktu luang.

Skenario Testing

Pada method Getter, dibentuk dahulu objek kelas tersebut. Lalu, hasil yang diharapkan dimasukkan ke suatu variable lalu dicocokkan dengan hasil Getter yang di-test.

Pada method Setter, dibentuk dahulu objek kelas tersebut. Lalu, nilai peubah yang diinginkan dimasukkan ke suatu variable dan dilakukan setter dengan variable tersebut , dan dicocokkan variable peubah tersebut dengan hasil getter setelah dilakukan setter.

Pada method getter yang menghasilkan ArrayList, hasil getter tersebut ditampilkan ke layar satu per satu untuk pengecekan.

Pada method SaveDragon, dibentuk dahulu objek dari kelas UserDragon. Lalu, dieksekusi method SaveDragon tersebut. Pengecekan dilakukan dengan melihat file .xml yang dihasilkan.

Pada method LoadDragon, pengecekan dimulai dengan mengeksekusi method LoadDragon tersebut dengan parameter input nama dragon yang ingin di-load. Dilakukan pengecekan terhadap nama dari dragon yang dihasilkan dari LoadDragon dengan nama dragon sebagai parameter input sebelumnya.

Pada method UseConsumable, dibentuk dahulu objek dari kelas UserDragon dan inventory-nya diisi dengan suatu consumable. Lalu, dieksekusi method UseConsumable tersebut. Dilakukan pengecekan terhadap status dragon setelah UseConsumable dengan status dragon sebelumnya ditambah status dari consumable.

Pada method AddConsumable, dibentuk dahulu objek dari kelas UserDragon dan objek dari kelas Consumable. Lalu, dieksekusi AddConsumable dengan parameter input consumable yang dibentuk sebelumnya. Pengecekan dilakukan terhadap uang dragon sebelumnya dikurangi harga consumable dengan uang dragon setelah dieksekusi method tersebut. Pengecekan juga dilakukan terhadap inventory dragon setelah pengeksekusian method apakah ada consumable yang tadi.

Pada method Train, dibentuk objek UserDragon. Pengecekan dilakukan terhadap atribut-atribut, seperti experience, level, maxHealth, maxStamina, apakah hasil dari eksekusi atribut-atribut tersebut sama dengan perhitungan yang diinginkan.

Pada method GenerateEnemy, dibentuk objek UserDragon dan dilakukan pengeksekusian method. Karena method ini hanya berisi me-random atribut-atribut dragon, pengecekan dilakukan apakah atribut-atribut dari dragon hasil eksekusi method berada pada range yang diinginkan.

Pada method fight ada 3 jenis kemungkinan (win, lose, draw) . Masing-masing kemungkinan tersebut dilakukan pengecekan tersendiri. Pada awalnya dibentuk objek Dragon dan objek UserDragon yang memiliki atribut-atribut memungkinkan untuk terjadi kemungkinan masing-masing. Dilakukan pengecekan terhadap atribut experience dan stamina, apakah mengalami perubahan dengan perhitungan yang diinginkan. Khusus untuk kasus win, dilakukan pengecekan atribut money juga, apakah mengalami perubahan pada range yang diinginkan.

Pada method Entertain, dibentuk objek UserDragon dan melakukan eksekusi method. Pengecekan dilakukan terhadap atribut happiness, apakah mengalami perubahan dengan perhitungan yang diinginkan.

Pada method Rest, dibentuk objek UserDragon dan melakukan eksekusi method. Pengecekan dilakukan terhadap atribut stamina dan health, apakah sudah mencapai max.

Pada method toToilet, dibentuk objek UserDragon dan melakukan eksekusi method. Pengecekan dilakukan terhadap atribut bladder, apakah sudah mencapai nol.

Pada method randomInt, tentukan nilai min, kemudian tentukan nilai max nilai max >= min, kemudian mengecek apakah range hasil random berada di antara min dan max

Pada method gaussBasedOn, tentukan nilai yang >= 0, kemudian cek apakah keluaran berada di range Math.ceil(0.8x) … Math.ceil(1.2x).

Deskripsi umum aplikasi

Versi-1:

Aplikasi yang kami buat seperti tamagotchi dengan hewan peliharaannya adalah naga. User hanya dapat memiliki satu naga. Naga tersebut bisa diajak bermain dengan pergerakan mouse user, disuruh bertarung dengan naga lain, berlatih, istirahat, dan pergi ke toilet. Aktivitas-aktivitas tersebut mempengaruhi parameter-parameter yang ada pada naga seperti, thirst, hunger, happiness, bladder, experience, level, money, health, stamina. Beberapa aktivitas seperti train dan fight dapat menaikkan level naga yang mempengaruhi gambar naga. Selain itu, seiring berjalannya waktu akan merubah parameter-parameter yang ada pula.

Versi-2:

Setelah koding, aplikasi yang kami buat memiliki perbedaan dari rencana semula. Namun perbedaan yang kami aplikasikan tidak mengubah struktur game secara keseluruhan hanya beberapa method saja. Contoh seperti cara mengajak bermain naga tidak menggunakan pergerakan mouse user, melainkan hanya meng-klik tombol “entertain” status naga akan langsung berubah. Kenaikan level sudah tidak mempengaruhi gambar naga. Kami juga tidak memasang fitur reset semua status pada game.

Workflow aplikasi

Pertamakali saat memasuki program, user akan ditampilkan form untuk login. User dapat mengisi nama dan password. User dapat memilih tindakan selanjutnya , yaitu create new atau login dengan dragon yang sudah ada.   
  
Setelah itu akan masuk ke dalam main program, yang akan ditampilkan status dragon saat itu. User bisa memilih untuk melakukan beberapa perbuatan kepada dragon, seperti memberi makan, menyuruh tidur, menyuruh latihan, ke toilet, istirahat, membeli barang ,dll.  
  
Jika user ingin menyelesaikan game, user dapat memilih untuk save and quit dan program akan menyimpan status dragon .

Non functional requirement

- Program dibuat tidak melanggar peraturan Undang Undang   
- Program di-release dengan Lisensi GPLv3  
- Resolusi game adalah minimal 809x490 pixel pada versi swing

- Atribut-atribut dapat mengalami perubahan 5 detik sekali.

* Pertama jika hunger > 75 :

Happiness: -5

Health: -2.4

Hunger: +2

Thirst: +2

Jika hunger > 50 :

Happiness: -3

Health: -1.2

* Kedua jika thirst > 75 :

Happiness: -5

Health: -2.4

Jika thirst > 50 :

Happiness: -3

Health: -1.2

* Ketiga jika hunger < 20 :

Bladder: +2

Jika hunger < 40 :

Bladder: +1.6

Jika hunger < 80 :

Bladder: +0.8

Lainnya:

Bladder: +0.4

* Keempat jika thirst < 20 :

Bladder: +2

Jika thirst < 40 :

Bladder: +1.6

Jika thirst < 80 :

Bladder: +0.8

Lainnya:

Bladder: +0.4

* Kelima jika bladder > 75 :

Happiness: -5

Health: -2.4

Jika bladder > 50 :

Happiness: -3

Health: -1.2

* Keenam jika happiness < 50 :

Health: -2.4

Jika happiness < 25 :

Health: -1.2

* Selain itu yang pasti terjadi:

Happiness: -random(5,10)/10

Health: random(3,7)

Stamina: random(5,10)/10

Money: random(1,5)

Thirst: random(5,20)/10

Hunger: random(5,20)/10

- Aksi entertain menambah happiness sebanyak 40 dengan maksimal happiness 100.

- Aksi rest menambah stamina dan health sampai maksimal .

- Aksi toToilet mengurangi bladder sehingga habis

- Aksi fight dapat mengurangi health dan stamina dan menambah experience point. Jika menang menambahkan uang.

* Jika menang :

Health: -10\*level

experience: +200+30\*level

money: +100+random()\*400

* Jika seri:

Health: -20\*level

Experience: +25\*level

* Jika kalah:

Health: -30\*level

Experience: +20\*level

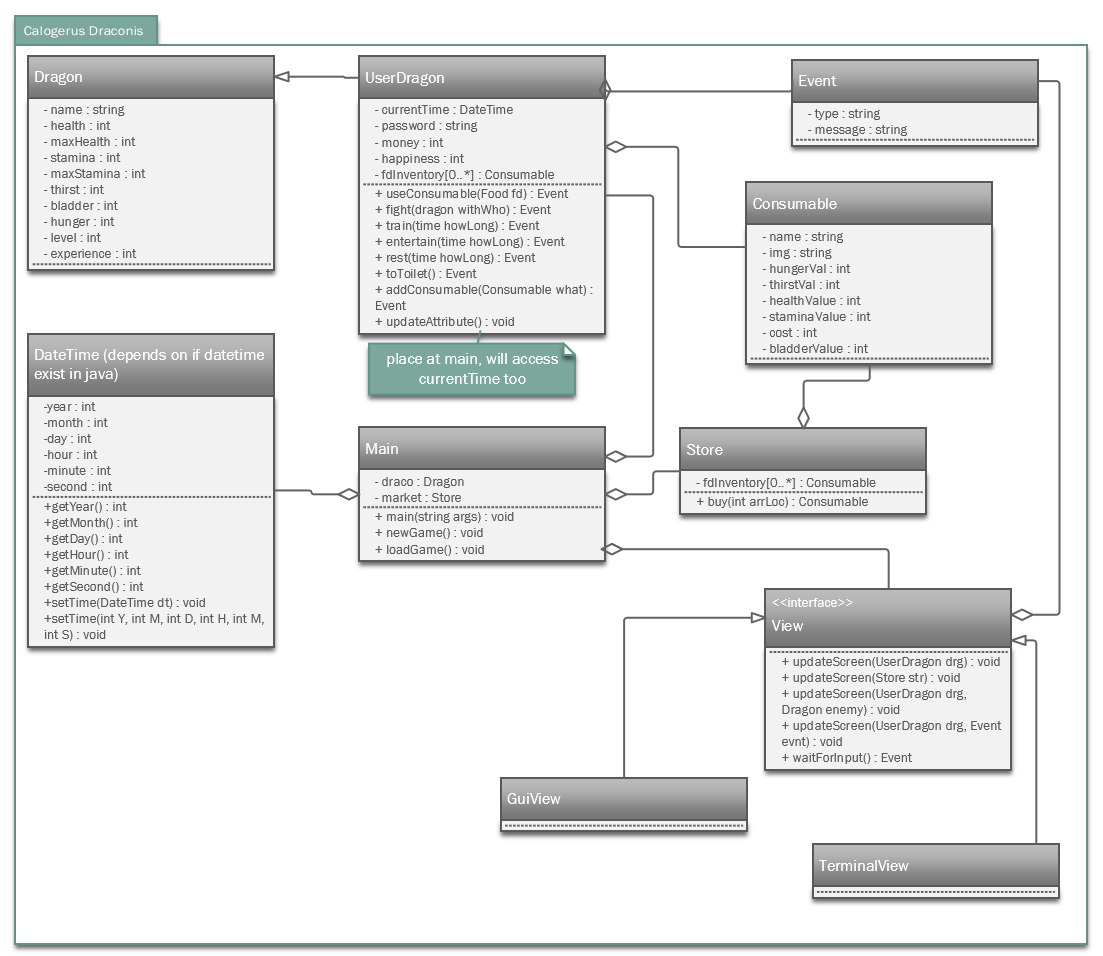
* Yang pasti terjadi:

Stamina: -20\*level

- Aksi train dapat menambah experience point sebanyak 20\*level.

- Jika experience yang telah diakumulasi mencapai batas experience yang dibutuhkan untuk naik level maka dragon akan naik level, serta menambah max health sebanyak 30 dan max stamina sebanyak 10.

Diagram Kelas (analisis)



Platform dan Tools

Program kami dibuat menggunakan NetBeans IDE 7.4 dan 8.0 .

Deskripsi singkat

1. Project Management

Rencana pengerjaan :

|  |  |
| --- | --- |
| Waktu | Rencana |
| 10 April – 13 April | Membuat Proposal v.0 |
| 14 April – 20 April | Membuat Diagram Kelas dan Functional Requirement |
| 21 April – 27 April | Menyelesaikan Non Functional Requirement |
| 28 April – 4 Mei | Membuat Kelas dalam Java |
| 5 Mei – 11 Mei | Membuat Terminal dan GUI |
| 12 Mei – 18 Mei | Membuat GUI |

Pembagian pekerjaan :

Rita : Membuat gambar dragon dan membuat gambar consumables, junit.

Willy : Membuat kelas Dragon,UserDragon , Event, Consumable , Store , TerminalView.

Denny : Membuat nonfunctional requirements seperti levelling experience , pengaturan attribut Dragon, attribute change per level , menentukan consumables dan atribut consumables.

Dariel : Membuat GuiView

Cilvia : Membuat method fight dan method updateAttribute pada kelas Dragon, junit.

Target product / deliverables : Source code yang memiliki 2 kelas view , terminal dan GUI.

1. Quality Management

Penamaan variable sesuai standard.

1. Configuration Management :
2. Base line : java swing , NetBeans IDE , terminal , laptop, internet, git.
3. Versioning :

Daftar Files :

Versi GUI:

GUIView.java

GUIViewFight.java

GUIViewLogin.java

InventoryDisplay.java

StoreDisplay.java

Versi Console :

TerminalView.java

Keduanya :

Consumables.java

Dragon.java

Event.java

Main.java

NotEnoughMoneyException.java

Store.java

UserDragon.java

XmlController.java

View.java

util.java

Build Script :

ant -f D:\\Tgs\\Sem4\\Oop\\if2210-tubes2 -Dnb.internal.action.name=run run

init:

Deleting: D:\Tgs\Sem4\Oop\if2210-tubes2\build\built-jar.properties

deps-jar:

Updating property file: D:\Tgs\Sem4\Oop\if2210-tubes2\build\built-jar.properties

compile:

run:

BUILD SUCCESSFUL (total time: 11 seconds)

Bug tracking:

Dilakukan dengan menggunakan exception handling dan debug mode di NetBeans IDE.