

MICHAEL VS. LALAPAN

Belimbingsquash

Kelompok 03





Game Description

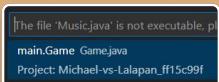
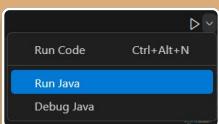


Michael vs Lalapan merupakan permainan versi sederhana dari permainan populer Plant vs. Zombies. Tujuan dari permainan ini adalah melindungi si pemilik rumah dari *zombies* menggunakan tanaman yang tersedia. Setiap *plants* dan *zombies* memiliki atribut dan kemampuan yang berbeda - beda. Pemain harus menanam *plants* yang tepat dan sesuai untuk melindungi rumah dari zombie.



How to Install

1. **Clone** repository GitHub
2. Buka dalam folder **Michael-vs-Lalapan**
3. Buka folder menggunakan IDE seperti **Visual Studio Code**
4. Klik tombol **Run Java** pada bagian pojok kanan atas
5. Pilih **Game.java** dan game akan berjalan



How to Play



Untuk memainkan permainan **Michael vs Lalapan**, gunakan **arrow key** untuk memindahkan **arrow cursor** ke atas, bawah, kanan, dan kiri. Gunakan **keypad 1 - 6** untuk menanam tanaman sesuai dengan urutan pada deck yang tampil pada bagian atas layar dan **keypad d** untuk menghapus tanaman.

Gameplay



Main Menu



Setelah program dijalankan, pemain akan langsung masuk ke tampilan menu utama. Pada menu tersedia beberapa opsi berupa start, plant list, zombie list, dan Exit, dan instruction.



Plants List



Pemain dapat melihat atribut dan kemampuan dari setiap plants dengan memilih opsi **Plants List** pada menu utama.



Zombies List



Pemain juga dapat melihat atribut dan kemampuan dari setiap zombies dengan memilih opsi **Zombies List** pada menu utama.



Help



Pada opsi kanan bawah juga terdapat **Help** yang berisi instruksi permainan.





Inventory & Deck



Saat pemain memilih opsi **start** pada main menu maka pemain akan masuk ke tampilan **preparation**. Pada mode ini pemain akan memilih 6 *plants* dari 10 *plants* yang tersedia untuk digunakan dalam permainan.



Inventory & Deck Buttons



Pemain dapat menukar antar *plants* yang berada di *deck* atau di *inventory* dengan **swap**. Pemain juga dapat menghapus semua *plants* di *deck* dengan menggunakan **clear**. Setelah selesai, pemain dapat memulai permainan dengan menekan **start**.





Gameplay



The Game



Ketika permainan dimulai, **deck** yang telah dipilih pemain akan ditampilkan. Pemain dapat meletakkan **plant** pada **kotak** yang tersedia sesuai dengan **sun** yang dibutuhkan.



Deck



Sun didapatkan melalui tanaman **Sunflower** dan ketika pagi hari (**25 Sun**) untuk setiap interval waktu yang acak antara **5 - 10 detik**. Setiap **plants** memiliki kemampuan serta **cooldown** yang berbeda-beda. Beberapa **zombie** juga ada yang memiliki kemampuan sehingga akan menghasilkan berbagai kondisi ketika bertemu dengan **plants**. Terakhir ada **icon sekop** yang dapat digunakan untuk menghapus tanaman.





Gameplay



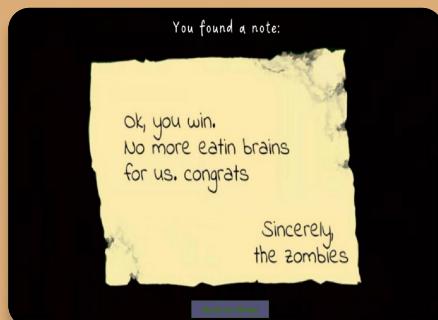
Losing Condition



Ketika **zombies** berhasil melewati *plants* dan **memasuki rumah** maka **game over**, pemain kalah dan permainan berakhir. Pemain bisa memiliki **Try Again** untuk mencoba bermain ulang atau **Quit** untuk kembali ke Menu.



Winning Condition



Ketika pemain berhasil mengalahkan semua *zombies*, akan muncul sebuah **surat** yang jatuh dari tubuh *zombie* terakhir. Ketika player melakukan **click** pada **surat** tersebut, permainan akan berakhir menandakan player **memenangkan** permainan.





Log Activity



TubEZZ.. tubEz..

Begadang yEs.. tidur nO

(Gambaran Mahasiswa STI H-2 Deadline Tubes)



18222118 Rizqi Andhika Pratama



CODING:

- Membuat **GitHub Repository**
- Membuat tampilan **GUI** pada packages **scenes**
- Membuat mekanisme **interaksi** *Plants* dan *Zombies*
- Membuat **managers** untuk setiap entity dan object



DOKUMEN:

- Melakukan **design** Game dan Booklet
- Membuat **Class Diagram**



18222116 Jason Jahja



CODING:

- Membuat keseluruhan Scenes **Preparation**
- Membuat mekanisme **aksi** dan **tampilan** setiap zombies
- Membuat **mekanisme kerja** kelas Sun
- Melakukan sebagian **debugging** program

DOKUMEN:

- Membantu membuat **Class Diagram**





Log Activity



18222133 Hanan Fitra Salam

CODING:

- Membuat semua **class Plants**

DOKUMEN:

- Mencari **resource image** dan **assets** untuk game
- Melakukan dan merancang **design** Booklet



18222121 Gymnastiar Anwar



CODING:

- Membuat semua **class Zombies**

DOKUMEN:

- Mencari **resource image, gif**, dan **assets** untuk game
- Melakukan **design** Booklet



18222129 Muhammad Faishal Putra

CODING:

- Membuat **class Victory** dan **GameOver**

DOKUMEN:

- Mencari **resource image, gif**, dan **assets** untuk game
- Melakukan **design** Booklet





Development Process - Milestone 1



Debut Meet-Up Satu Kelompok



Pertemuan pertama kami untuk membahas terkait pembagian tugas, perancangan *class diagram*, dan mengikuti asistensi bersama di **Nomar Kopi**, Jatinangor.



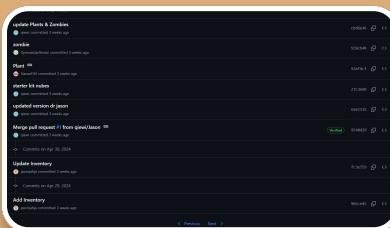
Asistensi Satu



Asistensi pertama dilakukan di **Nomar Kopi**, membahas *class diagram*, *design pattern*, dan skema code. Waktu asistensi lumayan berisik suara motor dari jalan raya.



Inisialisasi GitHub



Pembuatan repo dilakukan pada **27 April 2024**. Kemudian dilakukan pembuatan *class* yang sekiranya mudah sebagai permulaan sebelum menghadapi kenyataan.



Cari Resource Ini-Itu



#BingungMulainyaGimana - 2 minggu kebuang nyar referensi dan resource





Progress Sekian Lama



Setelah 2 minggu **no progress**, akhirnya coding dimulai **H-9 deadline** dan progress berlangsung **non-stop** setiap hari sampai dengan hari **H deadline**.



Asistensi Dua



Setelah menempuh sekian **banyak progress**, dilakukan asistensi kedua yang membawa hasil berupa **perbaikan** pada beberapa mekanisme program (terutama **Threading**).



Meet-Up Lagi Demi Nubes



Pertemuan kedua kami untuk membahas terkait progress tugas, menemukan banyak **bug**, dan memulai desain booklet di **Nomar Kopi**, Jatinangor.



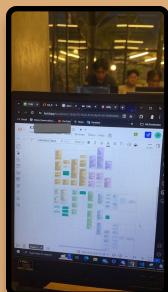
Hari-Hari Nemu Bug

Menjelang **H-4 deadline**, setiap hari dipenuhi oleh **bug** dan **debugging**. Ada yang squashnya balik ke rumah, ada yang zombie jumbo, intinya '**nguli**'.





Apa Itu Tidur?



Ketika **game**-nya sudah berjalan, hal yang sangat ‘**nguli**’ adalah membuat **class diagram**-nya. Dimulai dari **22.00 WIB**, pengeraian **class diagram** baru usai pukul **06.00 WIB**. Yha, saat-saat ketika **H-2 deadline**.

Besar harapan semoga penulis **sehat selalu** hingga sampai pada UAS. (ironis karena waktu menunjukkan pukul 00.00 WIB saat menulis ini)



Begadang B-nya Booklet!

Yes! begadang lagi dan kali ini alasannya ada mengerjakan **booklet**. Mau gimana lagi.. udah **hari H deadline** banget.



Akhirnya Bisa Tidur



Akhirnya, sampai juga di **hari H**, meet-up di **Nomar Kopi** (lagi) makasih banyak Nomar udah jadi tempat kita nubes selama ini.. booklet selesai, code selesai, class diagram juga. See you another time OOP, **BelimbingSquash** pamit izin turu.

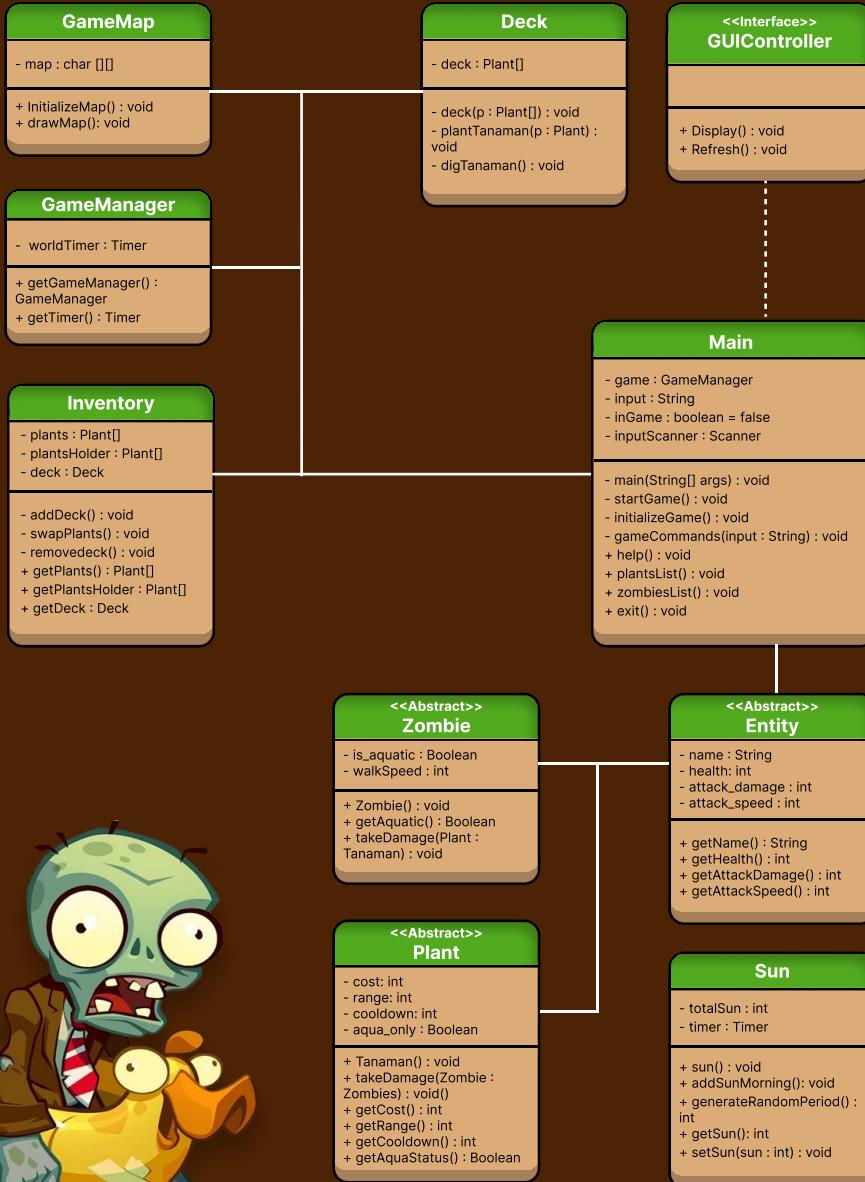




Class Diagram Awal

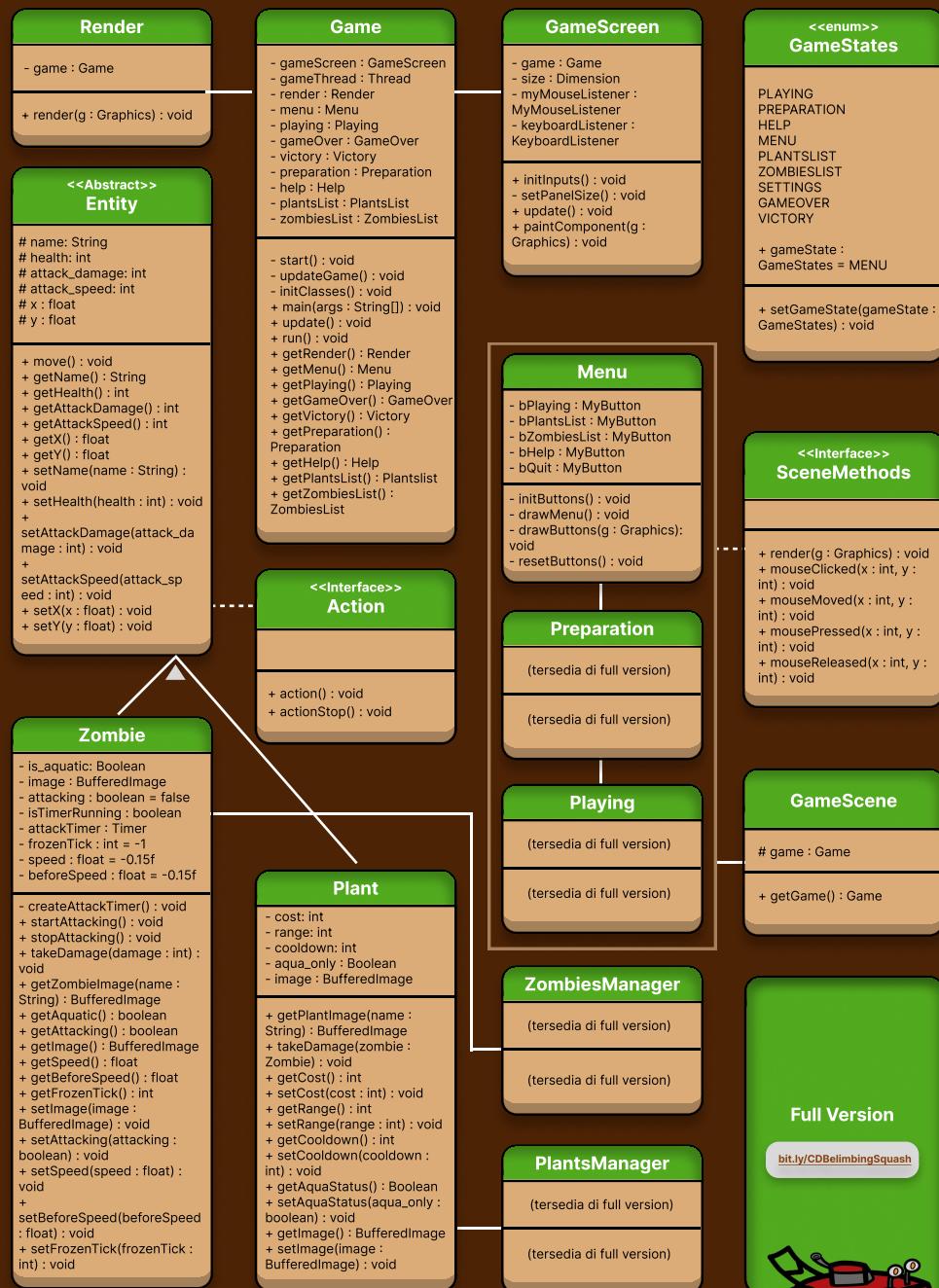
-

X





Class Diagram Akhir





Factory

Factory adalah design pattern creational yang bertujuan untuk **memisahkan proses pembuatan objek dari kode yang menggunakan objek tersebut**. Design pattern ini menyediakan antarmuka **untuk membuat objek dalam superklas**, namun memungkinkan subkelas untuk mengubah jenis objek yang akan dibuat.



State

State adalah design pattern behavioral yang memungkinkan **sebuah objek mengubah perilakunya ketika keadaan internalnya berubah**. Design pattern ini akan **terlihat seperti objek tersebut mengubah kelasnya**.



Iterator

Iterator adalah design pattern behavioral yang **memungkinkan untuk menelusuri elemen koleksi** tanpa memperlihatkan representasi dasarnya. Design pattern ini menyediakan cara untuk **mengakses elemen dalam sebuah koleksi** secara berurutan **tanpa harus mengetahui detail implementasi** dari koleksi tersebut.



Command

Command adalah design pattern behavioral yang **mengubah permintaan menjadi objek** yang berdiri sendiri **yang berisi semua informasi tentang permintaan tersebut**. Transformasi ini memungkinkan untuk meneruskan permintaan sebagai argumen metode, **menunda atau mengantari eksekusi permintaan**, dan **mendukung operasi yang dapat dibatalkan**.



Observer

Observer adalah design pattern behavioral yang memungkinkan untuk menentukan mekanisme berlangganan untuk memberi tahu banyak objek tentang peristiwa apa pun yang terjadi pada objek yang mereka amati. Design pattern ini **memungkinkan sebuah objek secara otomatis memberi tahu objek lain tentang perubahan statusnya**.



Mediator

Mediator adalah design pattern yang memungkinkan untuk **mengurangi ketergantungan yang kacau antar objek**. Design pattern ini **membatasi komunikasi langsung antar objek** dan **memaksa mereka untuk berkolaborasi hanya melalui objek mediator**, ini membantu dalam mengurangi keterikatan antar objek.





99+

TUBES OOP (Kelompok 3) (5)



Michael vs Lalapan

Jadi, gimana nih gais kesan pesan nubesnya?

Qie



Makasih banyak Nomar, Kopi Toleransi, dan Teman2

Hanan



Kenangan yang sementara, tapi ingin diulang, makasih gais

Michael vs Lalapan

Kalo yang lain gimana nih? sama juga nggak

Jason



Tubesnya seru, tapi lebih seru karena bareng <3

Gigim



Banyak keseruan yang tidak bisa dungkapkan

Faishal



Makasih bgt buat kalian semua yg sudah ada

