PROJECT UAS

Oleh kelompok :

1. *Arifin* *(2211102441102)*
2. *Adma Anafasha* *(2211102441126)*
3. *Ahya Alfarezi* *(2211102441115)*

Teknik Informatika Sains & Teknologi
Universitas Muhammadiyah Kalimantan Timur

Samarinda, 2023

PENDAHULUAN

Game "Jetpack Joyride" telah menjadi salah satu game mobile yang sangat populer sejak dirilis oleh Halfbrick Studios pada tahun 2011. Dengan mekanika permainan yang sederhana namun adiktif serta desain grafis yang menarik, game ini berhasil menarik perhatian jutaan pemain di seluruh dunia.

Laporan ini merupakan dokumentasi dari proyek pengembangan game versi adaptasi Jetpack Joyride yang dilakukan dalam lingkup platform tertentu. Proyek ini bertujuan untuk mereplikasi sebagian besar mekanika permainan, mengadaptasi elemen kunci dari versi aslinya, serta menambahkan sentuhan kreatif yang unik sesuai dengan lingkungan pengembangan yang digunakan.

Dalam pendahuluan ini, akan dijelaskan tujuan dari proyek pengembangan game ini, metodologi yang digunakan dalam proses pengembangan, serta bagaimana laporan ini akan membahas setiap langkah yang diambil selama proses pembuatan game ini.

TUJUAN PROJECT

Tujuan utama dari proyek ini adalah untuk mengembangkan versi adaptasi dari game Jetpack Joyride dalam suatu lingkungan pengembangan tertentu (misalnya, Greenfoot). Melalui proyek ini, diharapkan dapat mempelajari dasar-dasar pengembangan game, mengaplikasikan pengetahuan yang diperoleh dalam menciptakan mekanika permainan yang mirip dengan versi aslinya, serta mengeksplorasi kreativitas dalam menambahkan elemen baru ke dalam permainan.

METODOLOGI

Proyek pengembangan ini melibatkan beberapa tahapan utama, termasuk fase perencanaan, desain, pengkodean, pengujian, dan iterasi. Metodologi yang digunakan mencakup penggunaan platform tertentu, pemilihan algoritma, integrasi fitur-fitur kunci dari Jetpack Joyride, serta penyesuaian terhadap lingkungan pengembangan yang digunakan.

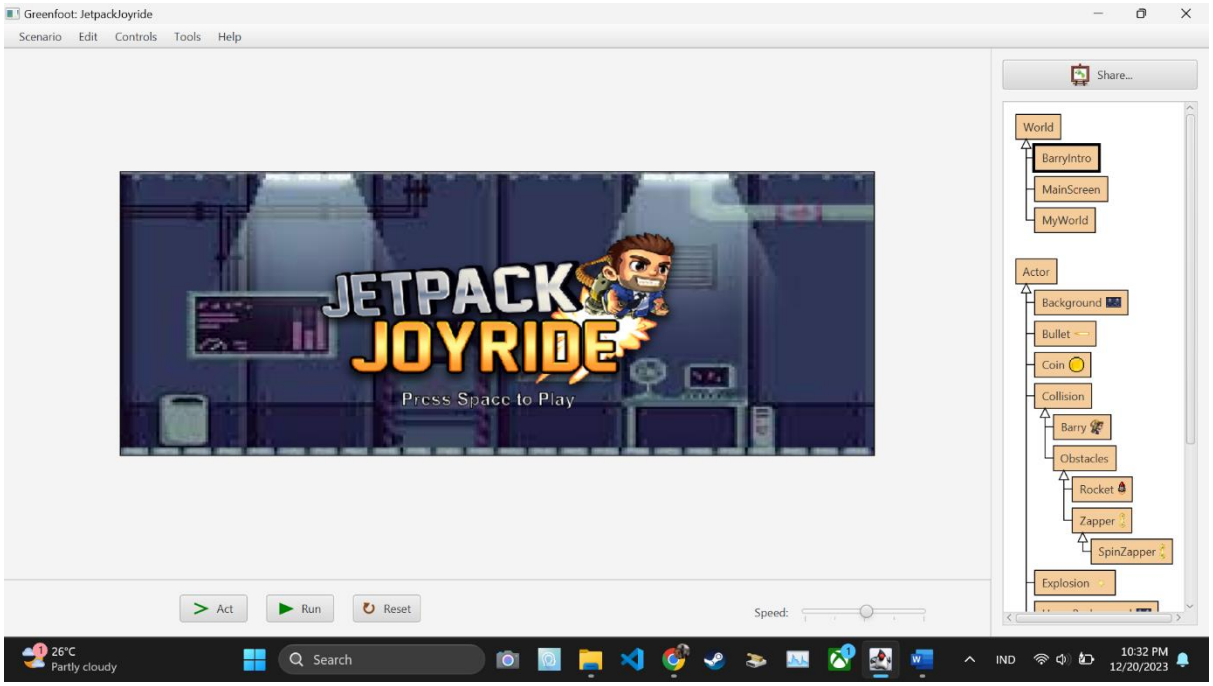
RANGKAIAN LAPORAN

Laporan ini akan menyajikan detail setiap tahap dalam proses pengembangan game Jetpack Joyride, termasuk tantangan yang dihadapi, solusi yang diambil, pembelajaran yang diperoleh, serta kesimpulan dari keseluruhan proyek.

Dengan memahami konteks ini, laporan ini diharapkan dapat memberikan gambaran komprehensif tentang proses pengembangan game adaptasi Jetpack Joyride serta memberikan wawasan yang berguna bagi pengembang game pemula atau mereka yang tertarik dalam pemrograman game.

PENJELASAN CODE PADA GAME

Untuk code pada game ini kami menerapkan banyak code yang kami dapatkan dari website greenfoot dan yang telah kami pelajari pada perkuliahan. Berikut ini adalah penjelasan code pada setiap subclass actor dan world di game Jetpack Joyride yang telah kami buat.



1. **World**
   a. BarryIntro :
      - super(800, 300, 1, false); : Membuat dunia baru dengan ukuran 800x300 sel dengan ukuran sel 1x1 piksel.
      - HomeBackground bk = new HomeBackground(); : Membuat objek HomeBackground.
      - GreenfootImage back = bk.getImage(); : Mengambil gambar dari objek HomeBackground.
      - back.scale(800, 380); : Mengubah skala gambar menjadi 800x380 piksel.
      - bk.setImage(back);: Menetapkan gambar yang telah diubah skala ke objek HomeBackground.
      - addObject(bk, 400, 150); : Menambahkan objek HomeBackground ke dunia pada posisi (400, 150).
      - StartingBarry b = new StartingBarry(); : Membuat objek StartingBarry.
      - addObject(b, -20, 275); : Menambahkan objek StartingBarry ke dunia pada posisi (-20, 275).
      - Explosion e = new Explosion(); : Membuat objek Explosion.
      - addObject(e, 0, -200); : Menambahkan objek Explosion ke dunia pada posisi (0, -200).
      - HomeBackground.homeClicked = true; : Mengatur nilai homeClicked dalam kelas HomeBackground menjadi true.
      - Kode ini bertanggung jawab untuk membuat dunia permainan dengan karakter, latar belakang, dan elemen permainan lainnya yang diperlukan untuk memulai pengalaman permainan.

   b. MainScreen :
      - Konstruktor MainScreen :
        - Membuat dunia baru dengan ukuran 800x300 sel dengan ukuran sel 1x1 piksel.
        - Memanggil metode placeBackground() dengan parameter tertentu.
      - Metode placeBackground(int scaleX, int scaleY, int X, int Y):
        - Membuat objek HomeBackground.
        - Mengambil gambar dari objek HomeBackground dan mengubah skala gambar sesuai dengan nilai scaleX dan scaleY.
        - Menetapkan gambar yang telah diubah skala ke objek HomeBackground.
        - Membuat objek Icon.
        - Mengambil gambar dari objek Icon dan mengubah skala gambar sesuai dengan perbedaan antara scaleX - 400 dan scaleY - 200.
        - Menetapkan gambar yang telah diubah skala ke objek Icon.
        - Membuat objek StartingBarry.

- Menambahkan objek HomeBackground, Icon, StartingBarry, dan Explosion ke dunia pada posisi yang ditentukan.
- Menampilkan teks "Press Space to Play" pada koordinat (390, 240) di dunia.

c. MyWorld :

- Terdapat variabel-variabel seperti delay, increase, zapperStartY, points, count, coinCount, dan highScore yang digunakan untuk melacak skor, jumlah koin, dan konfigurasi lain dalam permainan.
- Konstruktor MyWorld menciptakan dunia permainan dengan ukuran 800x300 piksel dan memanggil metode setUp() untuk menginisialisasi permainan.
- Metode act() dipanggil secara terus-menerus oleh Greenfoot. Di dalamnya, skor diperbarui berdasarkan peningkatan waktu dan beberapa kondisi seperti pemain kehilangan nyawa (objek Barry kosong).
- Ada beberapa metode bantuan seperti placeBackground, updateScore, dan metode lain yang mengatur tampilan latar belakang, pembaruan skor, dan konfigurasi lainnya.
- Metode setUp() mengatur ulang kondisi awal permainan dengan menghapus objek-objek tertentu, menempatkan kembali latar belakang, menetapkan skor dan jumlah koin ke nol, menambahkan objek Barry, Zapper, EndScreen, ReplayButton, dan lainnya ke dunia permainan.

2. Actor

a. Background :

- Terdapat variabel speedX, increase, dan delay. speedX digunakan untuk menentukan kecepatan pergerakan latar belakang horizontal. increase dan delay digunakan untuk mengatur interval pergerakan latar belakang.
- Terdapat konstruktor yang seharusnya menginisialisasi nilai increase menjadi 0. Namun, ada kekeliruan penamaan; konstruktor seharusnya tidak memiliki tipe data balik (void) dan harus memiliki nama yang sama dengan kelas (Background), bukan public void Background().
- Metode act() dipanggil secara terus-menerus oleh Greenfoot dan berisi logika pergerakan latar belakang. speedX dihitung berdasarkan kecepatan Zapper dan perubahan skor. Lalu, latar belakang dipindahkan secara horizontal dan variabel increase diincrement untuk mengatur interval pergerakan.
- Metode moveSpeed() digunakan untuk memindahkan latar belakang berdasarkan kecepatan yang dihitung sebelumnya. Pemindahan latar belakang hanya terjadi jika objek Barry ada dalam dunia permainan.

- Metode respawn() bertujuan untuk mengatur respawn (pemulihan) posisi latar belakang ketika mencapai batas tertentu ke arah kiri layar. Jika posisi latar belakang sudah melewati batas tertentu, latar belakang akan dipindahkan ke posisi awal sebelah kanan layar untuk memberikan efek pergerakan tanpa batas.
- Terdapat metode setLocationB(int x, int y) yang digunakan untuk menetapkan posisi latar belakang dengan koordinat (x, y).

b. Bullet :

- Terdapat variabel bulletTurn yang digunakan untuk menentukan putaran acak peluru saat dibuat. Nilai putaran peluru dihitung secara acak antara 60 dan 120 derajat.
- Terdapat konstruktor Bullet() yang akan dipanggil saat menciptakan objek peluru. Konstruktor ini mengatur putaran peluru dengan memanggil fungsi turn() untuk memutar peluru sesuai dengan nilai bulletTurn yang sudah dihitung sebelumnya.
- Metode act() adalah metode yang dipanggil secara berulang oleh Greenfoot. Di dalamnya, peluru akan bergerak ke atas dengan kecepatan 15 piksel setiap pemanggilan melalui perintah move(15). Selain itu, ada pemanggilan ke metode delete() yang berfungsi untuk menghapus peluru saat posisi Y-nya mencapai atau melampaui batas tertentu (dalam hal ini, 280 piksel dari atas layar).
- Metode delete() bertanggung jawab untuk menghapus objek peluru dari dunia permainan ketika posisi Y peluru melewati batas tertentu. Dalam kasus ini, jika posisi Y peluru mencapai 280 piksel, peluru akan dihapus dari dunia permainan dengan menggunakan perintah getWorld().removeObject(this).

c. Coin :

- Terdapat beberapa variabel yang digunakan untuk mengatur animasi koin (gifDelay, delay, increase, timer, currentImage, images, right, isAlive).
- Konstruktor Coin() menginisialisasi beberapa variabel seperti currentImage, increase, timer, gifDelay, dan delay. Ini digunakan untuk mengatur animasi koin dan pengaturan lainnya.
- Metode act() dipanggil secara berulang oleh Greenfoot. Di dalamnya terdapat beberapa fungsi:
    - animation(): Mengatur animasi koin dengan mengubah gambarnya dari satu ke yang lain setiap beberapa saat.

Menggunakan array gambar (images) untuk animasi dan mengubah gambar koin dengan interval tertentu.

- collect(): Memeriksa apakah koin bersentuhan dengan objek Barry (pemain). Jika bersentuhan, koin dihapus dari dunia permainan dan jumlah koin (coinCount) di dunia permainan ditambah 1.
- moveSpeed(): Menggerakkan koin ke kiri dengan kecepatan yang dihitung berdasarkan kecepatan Zapper dan perubahan skor.
- delete(): Menghapus koin dari dunia permainan jika koin sudah keluar dari batas layar atau sudah terkumpulkan oleh pemain.

- Metode-metode ini berfungsi untuk mengatur perilaku koin dalam permainan, termasuk animasi, interaksi dengan pemain, dan pergerakan koin.

d. Collision :

- getTouchedObjects(Class clss): Metode ini mengembalikan daftar semua aktor yang bersentuhan dengan objek ini dari kelas yang ditentukan. Ini melakukan loop pada semua objek dalam dunia permainan dari kelas yang diberikan dan menguji jika objek tersebut bersentuhan dengan objek ini (berpotongan dan bersentuhan piksel).
- touch(Class clss): Metode ini mengembalikan true jika objek ini bersentuhan dengan setidaknya satu aktor dari kelas yang ditentukan.
- getOneTouchedObject(Class clss): Metode ini mengembalikan satu objek yang bersentuhan dengan objek ini dari kelas yang ditentukan. Jika ada lebih dari satu objek yang bersentuhan, metode ini hanya akan mengembalikan salah satu dari mereka.
- touch(Actor a_big): Metode utama yang melakukan deteksi tabrakan piksel demi piksel. Metode ini menggunakan perhitungan detil seperti rotasi, pergeseran, dan tabrakan piksel antara objek yang dimasukkan sebagai argumen dan objek ini.
  - Metode touch(Actor a_big) melakukan perhitungan berikut:
  - Menentukan ukuran yang lebih kecil antara kedua gambar dan membuat gambar yang akan digunakan untuk deteksi tabrakan.
  - Membuat gambar baru untuk kedua aktor dan memutar sesuai dengan rotasi masing-masing.
  - Membandingkan piksel demi piksel dari kedua gambar untuk mendeteksi apakah ada tabrakan. Jika ada piksel pada kedua gambar yang tidak transparan (alpha > 0) dan bersentuhan, itu dianggap sebagai tabrakan.

e. Barry :

- Terdapat beberapa variabel yang digunakan untuk mengatur pergerakan, animasi, serta kondisi dan logika permainan (velY, delayI, delayD, delay, increase, timer, coinCount, barryY, zapperY, summonRocket, summonSpin, gifDelay, gifTimer, currentImage, first, nextZap, originalX, images, images1, Steps, MachineGun).
- Konstruktor Barry() menginisialisasi sejumlah variabel yang akan digunakan dalam kelas ini, termasuk pengaturan kecepatan vertikal (velY), delay untuk perubahan kecepatan (delayI, delayD), serta variabel lain yang digunakan untuk mengontrol animasi.
- Metode act() dijalankan terus-menerus oleh Greenfoot. Di dalamnya terdapat logika untuk menggerakkan karakter berdasarkan input pemain, memanggil objek baru seperti RocketWarning dan Bullet, serta mengontrol penciptaan objek Zapper dan Coin dalam permainan.
- Metode checkKeyPress() adalah inti dari pengontrol pergerakan karakter berdasarkan input. Jika tombol spasi ditekan, karakter akan terbang ke atas, jika tidak, karakter akan jatuh ke bawah atau berlari di lantai.
- Metode spawnZapper() bertanggung jawab atas logika untuk menciptakan objek Zapper secara periodik dalam permainan dengan aturan tertentu. Jika kondisi tertentu terpenuhi, objek Zapper atau SpinZapper akan diciptakan.
- Metode coinSetupBox() digunakan untuk menempatkan objek Coin dalam kotak dalam permainan.
- Metode animationRunning() dan animationFlying() mengatur animasi karakter Barry saat sedang berlari atau terbang, bergantung pada kondisi karakter di dalam permainan.

f. Obstacles :

- Kelas ini mewarisi dari kelas Collision, yang berarti kelas Obstacles mewarisi semua metode dan fungsi dari kelas Collision terkait dengan deteksi tabrakan piksel demi piksel.
- Metode kill() digunakan untuk menghapus objek Barry jika terjadi sentuhan antara objek Barry dan objek rintangan. Jika objek rintangan bersentuhan dengan objek Barry, maka objek Barry akan dihapus dari dunia permainan.
- Metode delete() bertujuan untuk menghapus objek rintangan dari dunia permainan jika sudah melewati batas tertentu (dalam kasus ini, jika posisi X objek rintangan kurang dari atau sama dengan -70 piksel).
- Terdapat satu metode abstrak yaitu moveSpeed(). Metode ini dideklarasikan sebagai abstrak, yang berarti semua kelas turunan dari Obstacles wajib mengimplementasikan metode ini. Metode ini memungkinkan setiap jenis objek rintangan untuk memiliki perilaku pergerakan khususnya sendiri,

tergantung dari jenis rintangan yang digambarkan. Implementasi sebenarnya dari moveSpeed() akan ditentukan di kelas turunannya.

g. Rocket
   - Kelas ini adalah turunan dari kelas Obstacles, yang berarti Rocket mewarisi metode dan sifat-sifat dari kelas Obstacles, seperti deteksi tabrakan piksel demi piksel dan metode kill() serta delete().
   - Pada konstruktor Rocket(), terdapat inisialisasi untuk properti-properti seperti orientasi putaran roket (turn(-90)), variabel delay untuk pergerakan (delay), variabel untuk mengatur animasi (gifDelay, timer, currentImage, increase), dan kecepatan horizontal (speedX) yang dihitung berdasarkan kecepatan Zapper dan perubahan skor dalam permainan.
   - Metode act() dijalankan terus-menerus oleh Greenfoot. Di dalamnya terdapat pemanggilan metode moveSpeed() untuk menggerakkan roket ke kiri, serta metode kill() dan delete() untuk mengatur interaksi antara roket dan objek lainnya (dalam hal ini, objek Barry). Selain itu, terdapat pemanggilan metode animation() untuk mengatur animasi roket.
   - Metode moveSpeed() bertanggung jawab untuk menggerakkan roket ke kiri dengan kecepatan yang dihitung berdasarkan kecepatan Zapper dan perubahan skor.
   - Metode animation() mengatur animasi roket dengan mengubah gambarnya dari satu ke yang lain setiap beberapa saat berdasarkan gifDelay. Ini dilakukan dengan mengubah gambar roket menggunakan array gambar (images) yang disediakan.

h. Zapper :
   - Kelas ini merupakan turunan dari kelas Obstacles, sehingga mewarisi fungsi-fungsi dasar yang diperlukan untuk objek rintangan, termasuk deteksi tabrakan (kill()) dan penghapusan objek jika melewati batas tertentu (delete()).
   - Pada konstruktor Zapper(), terdapat inisialisasi untuk properti-properti seperti kecepatan horizontal (velX), variabel delay untuk pergerakan (delay), variabel untuk mengatur animasi (gifDelay, gifTimer, currentImage, increase), serta inisialisasi rotasi (turn(angle)) berdasarkan nilai acak m. Rotasi ini digunakan untuk membuat variasi orientasi objek rintangan penghalang (Zapper) dalam permainan.
   - Metode act() dijalankan terus-menerus oleh Greenfoot. Di dalamnya, terdapat logika untuk animasi (animation()), deteksi pemain (Barry) yang masuk dalam area Zapper, dan pergerakan rintangan (moveSpeed()). Jika pemain berada dalam area rintangan (isTouching(Barry.class)), maka pemain akan dihapus dari dunia permainan.

- Metode kill() digunakan untuk menghapus objek Barry jika terjadi sentuhan antara objek Barry dan objek Zapper. Selain itu, akan dimainkan suara efek suara "ZapperDeath.mp3" untuk memberikan umpan balik audio kepada pemain.
- Metode moveSpeed() bertanggung jawab untuk menggerakkan objek Zapper ke kiri dengan kecepatan yang dihitung berdasarkan kecepatan velX, kecepatan Zapper, dan perubahan skor dalam permainan.
- Metode animation() mengatur animasi rintangan penghalang (Zapper) dengan mengubah gambarnya dari satu ke yang lain setiap beberapa saat berdasarkan gifDelay. Ini dilakukan dengan mengubah gambar rintangan menggunakan array gambar (images) yang disediakan.

i. SpinZapper
   - Pewarisan dan Inisialisasi:
     - Kelas SpinZapper merupakan turunan dari kelas Zapper, sehingga mewarisi semua properti dan metode yang ada pada Zapper.
     - Terdapat variabel once yang digunakan untuk mengontrol suatu peristiwa agar terjadi hanya sekali.
   - Metode act():
     - Metode act() merupakan metode yang dijalankan secara terus-menerus oleh Greenfoot.
     - Terdapat pemanggilan metode animation() yang mengatur animasi objek SpinZapper dengan mengubah gambarnya dari satu ke yang lain berdasarkan interval waktu tertentu.
     - Terdapat pemanggilan turn(2) yang mengatur perputaran objek SpinZapper sebesar 2 derajat setiap pemanggilan act().
     - Selanjutnya, terdapat kondisi if yang mengecek apakah objek pemain (Barry) ada di dalam dunia permainan. Jika pemain ada, maka akan dilakukan:
     - Peningkatan variabel increase yang digunakan untuk mengatur interval pergerakan.
     - Pemanggilan metode moveSpeed() yang mengatur pergerakan SpinZapper.
     - Deteksi dan reaksi terhadap sentuhan antara Barry dan SpinZapper melalui pemanggilan metode kill() dan delete().

j. Explosion :
- Properti Kelas:
  - Terdapat properti seperti increase yang digunakan untuk mengontrol interval animasi ledakan.
  - gifDelay adalah interval antara perubahan gambar ledakan.
  - currentImage digunakan untuk melacak indeks gambar yang sedang ditampilkan dalam animasi ledakan.
  - Array images menyimpan urutan gambar yang digunakan untuk animasi ledakan.
- Properti Statis Explosion:
  - Properti ini adalah objek GreenfootSound yang dipanggil saat objek Explosion diciptakan. Ini bertanggung jawab untuk memainkan suara ledakan pada saat objek Explosion dibuat.
- Metode act():
  - Metode act() akan dijalankan terus-menerus oleh Greenfoot.
  - Di dalamnya, terdapat kondisi yang memeriksa apakah HomeBackground.homeClicked bernilai true. Jika ya, maka:
    - setLocation(0, 250) digunakan untuk menempatkan ledakan pada koordinat (0, 250) di layar.
    - Pemanggilan metode animation() untuk memulai animasi ledakan.
    - Pemutaran suara ledakan menggunakan objek Explosion.
- Metode animation():
  - Metode ini mengatur animasi ledakan.
  - increase bertanggung jawab untuk mengontrol interval antara perubahan gambar ledakan.
  - Jika increase mencapai gifDelay, gambar ledakan akan diperbarui ke gambar selanjutnya dalam urutan.
  - Jika animasi mencapai gambar terakhir dalam array images, objek Explosion akan dihapus dari dunia permainan.

k. HomeBackground :
- Variabel dan Properti:
  - homeClicked: Variabel boolean statis yang mengontrol apakah latar belakang rumah telah diklik.
  - timer, delay, dan increase: Variabel yang digunakan untuk mengatur pengaturan waktu dan interval.
  - backgroundMusic dan backgroundMusic1: Objek GreenfootSound yang mewakili musik latar belakang yang akan dimainkan dalam latar belakang.
- Metode act():
  - Metode act() dipanggil terus-menerus oleh Greenfoot.

- Pertama, musik latar belakang (backgroundMusic1) akan diputar secara berulang menggunakan playLoop().
- Setelah itu, terdapat kondisi if yang mengecek apakah tombol "space" ditekan (Greenfoot.isKeyDown("space")). Jika iya, variabel homeClicked diatur menjadi true, dan semua objek dari kelas Icon dihapus dari dunia permainan.
- Jika homeClicked sudah bernilai true, langkah-langkah berikut dilakukan:
  - Musik latar belakang saat ini (backgroundMusic1) dihentikan.
  - Text yang ditampilkan di koordinat (390, 240) dihapus dari layar menggunakan getWorld().showText("", 390, 240).
  - Terdapat perhitungan waktu (timer) dengan menggunakan delay. Jika timer mencapai atau melebihi nilai delay, permainan akan beralih ke dunia MyWorld yang baru dengan Greenfoot.setWorld(new MyWorld()).
  - Selain itu, beberapa pengaturan lain dilakukan seperti menghentikan musik latar belakang yang sebelumnya dimainkan (backgroundMusic), serta menghentikan suara ledakan jika ada (Explosion.Explosion.stop()).

l. Icon :

- Dalam konteks kode yang disediakan, kelas ini tidak memiliki perilaku atau aksi yang didefinisikan dalam metode act(). Meskipun demikian, kelas ini secara konseptual mewakili objek yang ada di latar belakang menu utama. Kelas ini merupakan bagian dari struktur kelas dalam permainan dan dapat digunakan untuk menambahkan fungsionalitas atau perilaku tertentu untuk ikon atau objek tersebut di masa mendatang. Dengan adanya kelas ini, memungkinkan untuk mengembangkan perilaku yang spesifik terhadap ikon ini dengan menambahkan logika dalam metode act() atau menambahkan properti khusus dan metode lainnya sesuai kebutuhan permainan.

m. MainScreenButton :

- Variabel Statis mainButtonClicked:
  - Variabel boolean statis yang mengontrol apakah tombol utama sudah diklik atau tidak.
- Metode move():
  - Metode ini digunakan untuk menetapkan lokasi dari objek tombol ke koordinat (320, 210).

- Metode act():
    - Metode act() dijalankan terus-menerus oleh Greenfoot untuk menangani logika perilaku objek.
    - Pertama, ada pengecekan kondisi: Jika tidak ada objek Barry di dunia saat ini (diperiksa menggunakan getWorld().getObjects(Barry.class).isEmpty()), maka tombol akan dipindahkan ke posisi yang ditentukan menggunakan metode move().
    - Selanjutnya, ada kondisi yang memeriksa apakah tombol ini diklik oleh mouse (Greenfoot.mouseClicked(this)). Jika ya, maka:
        - Objek tombol (this) akan dihapus dari dunia permainan menggunakan getWorld().removeObject(this).
        - Dunia permainan akan beralih ke dunia MainScreen yang baru menggunakan Greenfoot.setWorld(new MainScreen()).

n. ReplayButton :
- Variabel Statis clicked:
    - Ini adalah variabel boolean statis yang mengontrol apakah tombol ulang permainan sudah diklik atau tidak.
- Metode move():
    - Metode ini menetapkan lokasi dari objek tombol ulang permainan ke koordinat (455, 210).
- Metode act():
    - Metode ini dieksekusi terus-menerus oleh Greenfoot untuk menangani logika perilaku objek.
    - Pertama, terdapat pengecekan kondisi: Jika tidak ada objek Barry di dunia saat ini (diperiksa menggunakan getWorld().getObjects(Barry.class).isEmpty()), maka tombol akan dipindahkan ke posisi yang ditentukan menggunakan metode move().
    - Kemudian, ada kondisi yang memeriksa apakah tombol ini diklik oleh mouse (Greenfoot.mouseClicked(this)). Jika iya, maka:
        - Dunia permainan akan beralih ke dunia BarryIntro yang baru menggunakan Greenfoot.setWorld(new BarryIntro()).
        - Objek tombol (this) akan dihapus dari dunia permainan menggunakan getWorld().removeObject(this).
- Metode Statis getClicked() dan setClicked(boolean b):
    - Metode statis ini digunakan untuk mendapatkan nilai dari variabel clicked dan mengatur nilainya. Metode getClicked() digunakan untuk mendapatkan nilai dari clicked, sedangkan

setClicked(boolean b) digunakan untuk mengatur nilai variabel clicked menjadi nilai yang diberikan sebagai argumen saat memanggilnya.

o. EndScreen :
- Metode move():
  - Mirip dengan yang ada di kelas ReplayButton, metode ini menetapkan lokasi dari objek layar akhir ke koordinat (400, 150).
- Metode act():
  - Metode ini dieksekusi terus-menerus oleh Greenfoot untuk menangani logika perilaku objek.
  - Pertama, ada pengecekan kondisi: Jika tidak ada objek Barry di dunia saat ini (diperiksa menggunakan getWorld().getObjects(Barry.class).isEmpty()), maka layar akhir akan dipindahkan ke posisi yang ditentukan menggunakan metode move().
  - Selanjutnya, terdapat kondisi yang memeriksa apakah tombol ulang permainan (clicked) sudah diklik. Jika sudah, maka posisi layar akhir akan dipindahkan ke lokasi yang tidak terlihat di luar batas layar permainan (x = 1000, y = 1000) menggunakan setLocation(1000, 1000). Ini memungkinkan layar akhir untuk menghilang dari layar permainan setelah tombol ulang permainan ditekan.
  - Terdapat dua kondisi tambahan: Jika layar akhir menyentuh objek Zapper atau Coin, maka objek tersebut akan dihapus dari layar akhir menggunakan removeTouching().

p. RocketWarning :
- Konstruktor RocketWarning(Barry b):
  - Menerima objek Barry sebagai parameter dan menetapkannya ke variabel b1.
  - Menetapkan nilai awal untuk beberapa variabel, termasuk delay, increase, dan first.
- Metode delete():
  - Menghapus objek RocketWarning dari dunia.
- Metode addedToWorld(World world):
  - Digunakan untuk mendapatkan referensi ke MyWorld tempat RocketWarning ditambahkan.
- Metode act():
  - Merupakan metode yang dieksekusi secara terus-menerus oleh Greenfoot untuk mengatur perilaku objek.

- Melakukan serangkaian pengecekan dan manipulasi posisi antara objek RocketWarning dan roket lainnya (Rocket).
- Jika objek Barry tidak kosong di dunia:
  - Terdapat serangkaian perbandingan posisi dan manipulasi terhadap objek RocketWarning dan roket lainnya berdasarkan posisi relatif mereka.
  - Jika kondisi tertentu terpenuhi (misalnya, terjadinya tumpang tindih antara dua objek RocketWarning), dilakukan penyesuaian posisi antara keduanya.
  - Pada saat tertentu (delay mencapai nilai tertentu), RocketWarning akan meluncurkan objek Rocket (r1) di posisi (800, getY()) dalam MyWorld.
  - Peringatan suara peluncuran dan suara misil dimainkan pada kondisi-kondisi tertentu.
- Jika tidak ada objek Barry di dunia, maka objek RocketWarning akan dihapus dari dunia dan first akan disetel ke false.

q. StartingBarry :

- Variabel Kelas:
  - images: Array yang berisi nama file gambar yang digunakan untuk animasi karakter Barry.
  - timer: Menghitung waktu sejak perubahan gambar terakhir.
  - gifDelay: Interval antara perubahan gambar dalam animasi.
  - currentImage: Menyimpan indeks gambar saat ini yang ditampilkan.
- Konstruktor:
  - Menetapkan nilai awal untuk variabel timer, gifDelay, dan currentImage.
- Metode act():
  - Memainkan animasi dengan memanggil metode animationRunning().
  - Jika tombol 'space' ditekan di HomeBackground, maka posisi StartingBarry bergeser ke kanan sebesar 3 piksel (setLocation(getX() + 3, getY())).
- Metode animationRunning():
  - Mengelola animasi dengan mengubah gambar yang ditampilkan setiap gifDelay frame.
  - Setiap gifDelay frame, gambar yang ditampilkan berganti dengan gambar berikutnya dalam array images.
  - Jika currentImage melebihi indeks gambar terakhir, kembali ke gambar pertama untuk membuat efek loop pada animasi.

KESIMPULAN

"Jetpack Joyride adalah permainan mobile yang seru di mana pemain mengendalikan karakter yang menggunakan jetpack untuk terbang melalui level-level yang berbeda. Permainan ini dirancang dengan baik dan menawarkan banyak tantangan menarik kepada para pemainnya.

Karakter utama dalam permainan ini adalah Barry, yang menggunakan jetpack untuk terbang melewati berbagai rintangan dan mengumpulkan koin sebanyak mungkin. Selama perjalanan, pemain harus menghindari berbagai rintangan seperti misil, laser, dan halangan lainnya.

Selain jetpack, terdapat juga berbagai power-up yang dapat membantu pemain dalam permainan, seperti jetpack yang lebih kuat, pelindung, atau item lain yang meningkatkan performa karakter.

Permainan ini memiliki mekanisme kontrol yang sederhana, yang membuatnya mudah dimainkan oleh berbagai kalangan. Grafiknya pun menarik dengan beragam tampilan yang kreatif dan level-level yang dirancang dengan baik.

Keseluruhan, Jetpack Joyride adalah permainan yang seru dan menarik bagi siapa pun yang menyukai tantangan, aksi, dan kecepatan."