

LAPORAN TUGAS AKHIR PEMROGRAMAN BERORIENTASI OBJEK

Game Car Wars

*Diajukan untuk memenuhi tugas mata kuliah Pemograman Berorientasi Objek yang diampu
oleh dosen: Sayekti Harist*



Disusun Oleh:

Calvin Saputra Pratama 2211102441032

Alif Ihsan 2211102441190

Rizaldi Darma 2211102244116

Dyah Fathonah Ramadhan 2211102441031

**UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR
SAMARINDA**

2023

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Allah SWT, yang telah melimpahkan rahmat, hidayah, dan karunia-Nya sehingga kami dapat menyelesaikan tugas akhir semester ini. Shalawat serta salam senantiasa tercurah kepada junjungan kita, Nabi Muhammad SAW, yang menjadi rahmat bagi seluruh alam.

Laporan ini disusun sebagai bagian dari tugas akhir semester dalam rangka pemenuhan mata kuliah Pemograman Berorientasi Objek pada semester 3 program studi Teknik Informatika. Topik yang kami angkat dalam makalah ini adalah "Pembuatan Game Menggunakan Greenfoot." Pembuatan game merupakan salah satu bidang yang sangat menarik dan relevan dengan perkembangan teknologi informasi saat ini.

Greenfoot dipilih sebagai alat pengembangan game dalam makalah ini karena platform ini memberikan pendekatan yang sangat baik bagi pemula yang ingin memahami konsep dasar pemrograman dan pengembangan game. Melalui pembahasan yang kami sajikan, diharapkan pembaca dapat memahami langkah-langkah dasar dalam menciptakan sebuah game sederhana menggunakan Greenfoot.

Pembuatan makalah ini tidak lepas dari bimbingan dan dukungan berbagai pihak yang turut serta dalam proses pembelajaran kami. Oleh karena itu, kami ingin mengucapkan terima kasih yang sebesar-besarnya kepada dosen pengampu mata kuliah Bapak Sayekti Harist yang telah memberikan bimbingan, arahan, dan masukan yang sangat berharga selama penyusunan makalah ini.

Tak lupa, kami juga mengucapkan terima kasih kepada teman-teman seangkatan yang telah saling memberikan dukungan, informasi, dan inspirasi selama proses pembelajaran. Semoga makalah ini dapat memberikan manfaat dan menjadi tambahan wawasan bagi pembaca yang tertarik dengan pengembangan game menggunakan Greenfoot.

Akhir kata, mohon maaf apabila terdapat kekurangan dalam penyusunan makalah ini. Kritik dan saran yang membangun sangat kami harapkan demi perbaikan di masa yang akan datang.

Semoga laporan ini dapat menjadi sumbangan kecil kami dalam mengembangkan pemahaman dan keterampilan di bidang pengembangan game. Terima kasih.

DAFTAR ISI

COVER HALAMAN.....	I
KATA PENGANTAR	II
BAB I.....	II
PENDAHULUAN	II
A. LATAR BELAKANG	III
B. TUJUAN.....	III
C. MANFAAT.....	III
BAB II	IV
PEMBAHASAN.....	IV
BAB III	V
KESIMPULAN.....	V

BAB I
PENDAHULUAN

Game menjadi bagian penting dalam dunia digital, memberikan pengalaman interaktif dan hiburan yang tak tertandingi. Dalam makalah ini, kami akan membahas perjalanan pembuatan game yang menggabungkan kecepatan, strategi, dan pertempuran antar-kendaraan, yang disebut Car Wars.

Car Wars bukan hanya permainan balap biasa; ini adalah dunia di mana pemain dapat merasakan sensasi mengendalikan mobil futuristik, bersaing dalam lintasan yang penuh tantangan, dan terlibat dalam pertempuran yang sengit. Makalah ini akan membahas setiap tahapan pembuatan Car Wars, dari ide awal hingga implementasi teknis yang menarik.

Dengan membahas ini, kami berharap dapat menjelajahi berbagai aspek kreatif, teknis, dan konseptual yang terlibat dalam menciptakan pengalaman gaming yang menghibur dan menarik. Bersiaplah untuk masuk ke dalam dunia Car Wars, di mana kecepatan dan strategi bergabung dalam satu petualangan yang tak terlupakan. Mari kita mulai mengungkap rahasia di balik layar pembuatan game ini!

A. LATAR BELAKANG

Pada era digital ini, game telah menjadi bagian tak terpisahkan dari kehidupan sehari-hari. Game bukan sekadar hiburan biasa, melainkan wadah di mana kreativitas dan teknologi bersatu. Dalam konteks ini, kita akan membahas dasar belakang di balik pembuatan game "Car Wars."

Dengan teknologi terus berkembang, pengembang game berusaha menciptakan pengalaman bermain yang lebih menarik. Game "Car Wars" muncul sebagai hasil dari upaya kreatif untuk menghadirkan pengalaman unik yang menggabungkan kecepatan, strategi, dan pertempuran antar-kendaraan.

Industri game selalu mencari inovasi, dan "Car Wars" adalah contoh nyata bagaimana kombinasi balapan dan pertempuran dapat menciptakan sesuatu yang spesial. Dalam dasar belakang ini, kita ingin melihat lebih dekat apa yang mendasari pembuatan game ini dan mengapa ini menjadi pilihan yang menarik.

Dengan memahami dasar belakangnya, diharapkan kita bisa lebih menghargai peran "Car Wars" dalam dunia game serta bagaimana game ini memberikan warna baru pada pengalaman bermain. Semoga penjelasan sederhana ini membantu kita memahami keunikan game "Car Wars" dengan lebih baik.

B. TUJUAN

Adapun tujuan penulisan dan pembuatan laporan ini adalah untuk memenuhi Tugas Semester Akhir dari matakuliah Pemograman Berorientasi Objek, pada Semester 3, dengan membahas proses pembuatan game Car Wars

C. MANFAAT

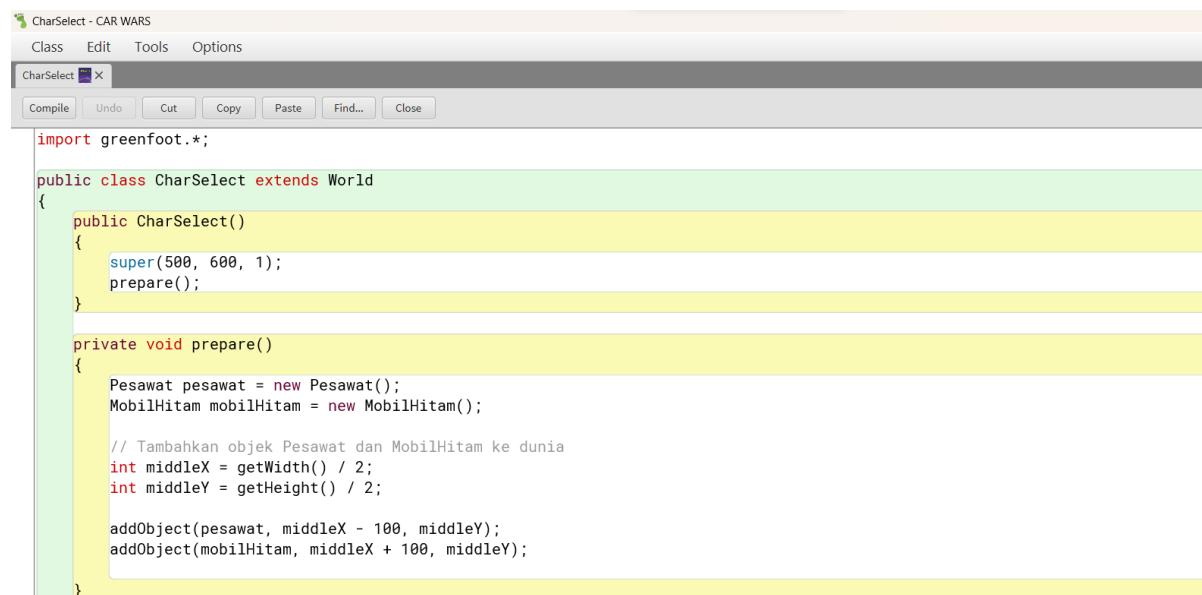
Dengan dibuatnya laporan ini diharapkan dapat memberi manfaat bagi pembaca khususnya mahasiswa yang terlibat dalam pembuatan laporan ini. Dari penulisan laporan ini pula diaharapkan mahasiswa mengambil manfaat seperti :

- proses pengembangan game
- mengimplementasikan GreenFoot
- memahami konsep serta elemen dalam pembuatan game di GreenFoot

BAB II

PEMBAHASAN

CharSelect



```
CharSelect - CAR WARS
Class Edit Tools Options
CharSelect X
Compile Undo Cut Copy Paste Find... Close
import greenfoot.*;
public class CharSelect extends World
{
    public CharSelect()
    {
        super(500, 600, 1);
        prepare();
    }
    private void prepare()
    {
        Pesawat pesawat = new Pesawat();
        MobilHitam mobilHitam = new MobilHitam();

        // Tambahkan objek Pesawat dan MobilHitam ke dunia
        int middleX = getWidth() / 2;
        int middleY = getHeight() / 2;

        addObject(pesawat, middleX - 100, middleY);
        addObject(mobilHitam, middleX + 100, middleY);
    }
}
```

Penjelasan:

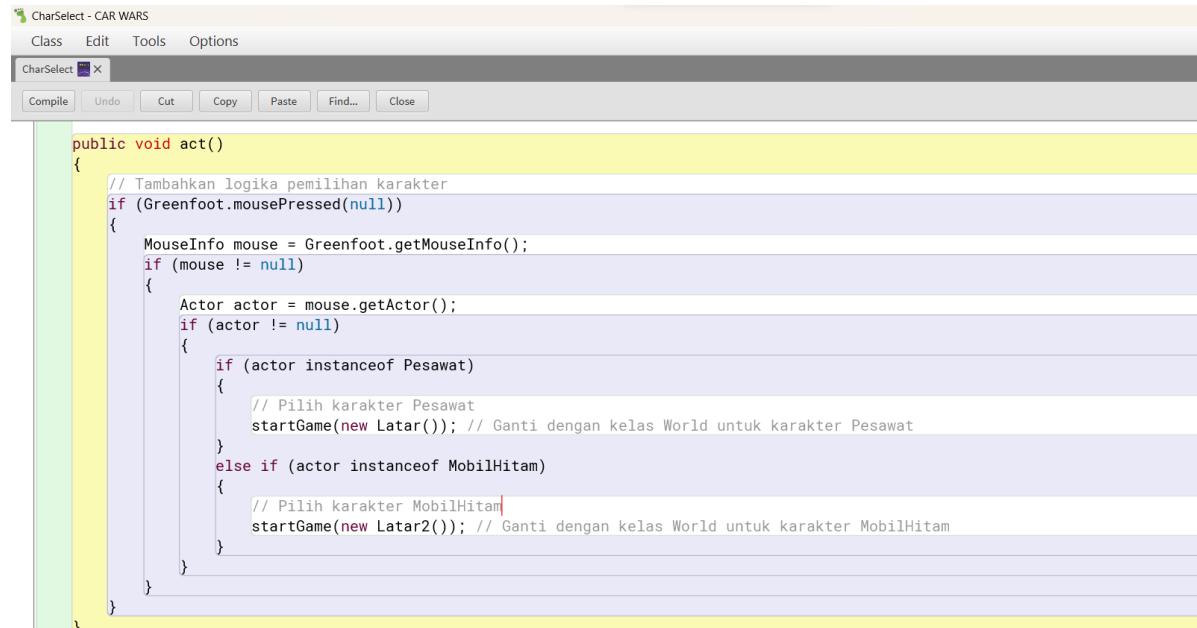
1. Constructor (`public CharSelect()`):

- Membuat dunia baru dengan ukuran 500x600 piksel.
- Memanggil method `prepare()` untuk menyiapkan dunia.

2. `prepare()` Method:

- Membuat objek Pesawat dan MobilHitam.
- Menambahkan objek ke tengah dunia dengan posisi relatif.

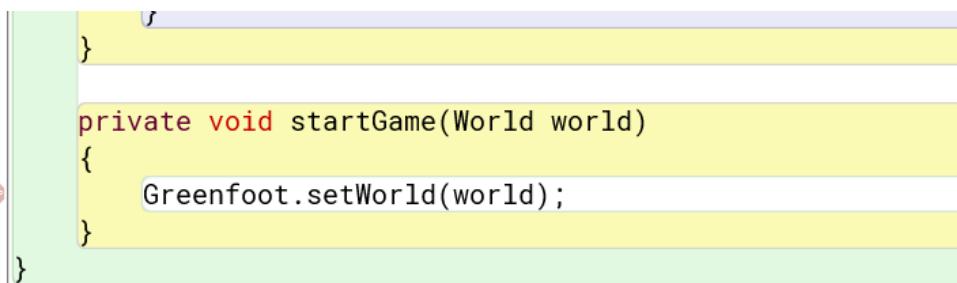
Latar



```
CharSelect - CAR WARS
Class Edit Tools Options
CharSelect X
Compile Undo Cut Copy Paste Find... Close
public void act()
{
    // Tambahkan logika pemilihan karakter
    if (Greenfoot.mousePressed(null))
    {
        MouseInfo mouse = Greenfoot.getMouseInfo();
        if (mouse != null)
        {
            Actor actor = mouse.getActor();
            if (actor != null)
            {
                if (actor instanceof Pesawat)
                {
                    // Pilih karakter Pesawat
                    startGame(new Latar()); // Ganti dengan kelas World untuk karakter Pesawat
                }
                else if (actor instanceof MobilHitam)
                {
                    // Pilih karakter MobilHitam
                    startGame(new Latar2()); // Ganti dengan kelas World untuk karakter MobilHitam
                }
            }
        }
    }
}
```

3. `act()` Method:

- jika mouse mengklik Peawat (Mobil Merah) maka akan memainkan game dengan class Pesawat(MobilMerah), lalu sebaliknya



```

    }

    private void startGame(World world)
    {
        Greenfoot.setWorld(world);
    }
}

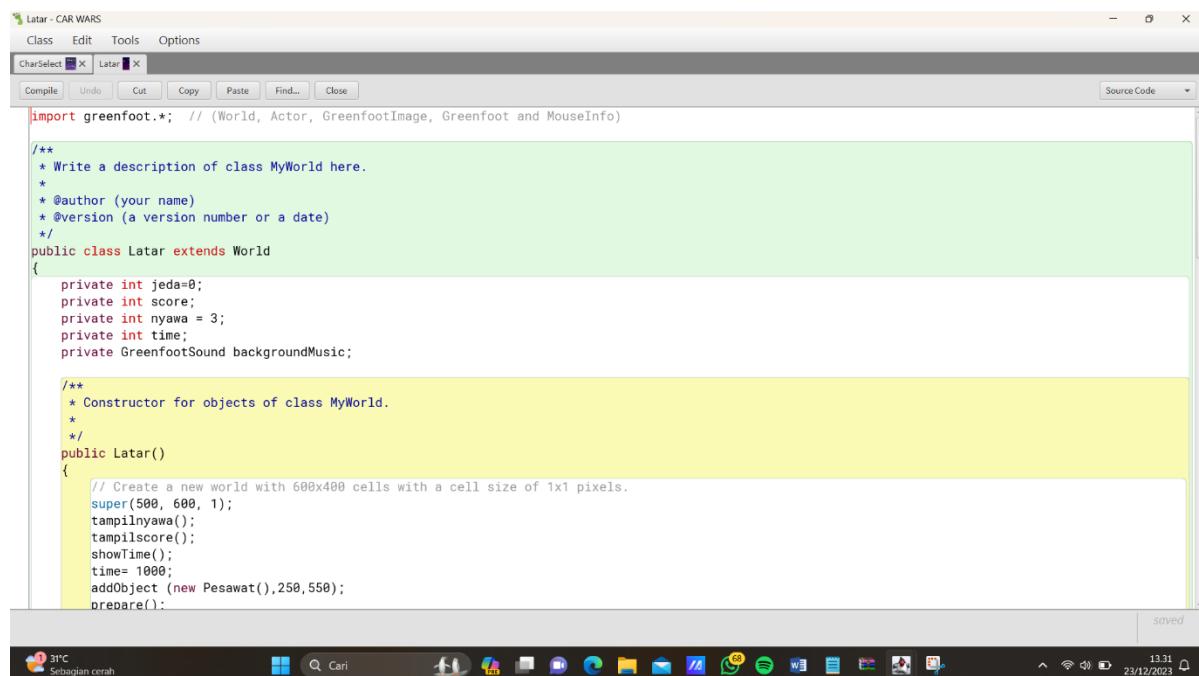
```

4. startGame() Method:

- Mengganti dunia saat ini dengan dunia baru yang sesuai dengan karakter yang dipilih.

Kelas ini bertanggung jawab untuk menampilkan pemilihan karakter dan mengarahkan pemain ke dunia yang sesuai.

Latar2



```

Latar - CAR WARS
Class Edit Tools Options
CharSelect X Latar X
Compile Undo Cut Copy Paste Find... Close Source Code
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/**
 * Write a description of class MyWorld here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Latar extends World
{
    private int jeda=0;
    private int score;
    private int nyawa = 3;
    private int time;
    private GreenfootSound backgroundMusic;

    /**
     * Constructor for objects of class MyWorld.
     */
    public Latar()
    {
        // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
        super(500, 400, 1);
        tampilnyawa();
        tampilscore();
        showTime();
        time= 1000;
        addObject (new Pesawat(),250,550);
        prepare();
    }
}

```

1. Constructor (`public Latar()`):

- Membuat dunia baru dengan ukuran yang tertera
- Menampilkan informasi nyawa, skor, dan waktu.
- Menambahkan objek Pesawat ke posisi awal.
- Memulai pemutaran musik latar menggunakan file audio "videoplayback.wav".

The screenshot shows the Greenfoot software interface with the title bar "Latar - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". Below the menu is a toolbar with buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu "Source Code" is open. The main area displays the Java code for the "Latar" class:

```
backgroundMusic = new GreenfootSound("videoplayback.wav");
backgroundMusic.playLoop();

public void updatenyawa(int point)
{
    nyawa = nyawa + point;
    tampilnyawa();

    if(nyawa == 0)
    {
        Greenfoot.stop();
        addObject(new kalah(), 250, 300);
    }
}

public void tampilnyawa()
{
    showText("Nyawa :" +nyawa, 400, 25);
}

public void addScore (int points){
    score = score + points;
    tampilscore();
}

private void tampilscore()
{
    showText ("Score :" +score, 100, 25);
}
```

2. updatenyawa Method:

- Mengupdate jumlah nyawa berdasarkan poin yang diberikan.
- Memanggil method `tampilnyawa()` untuk menampilkan informasi nyawa.
- Jika nyawa habis, menghentikan permainan dan menambahkan objek `kalah`.

3. tampilnyawa Method:

- Menampilkan jumlah nyawa pada posisi tertentu di layar.

4. addScore Method:

- Menambahkan skor berdasarkan poin yang didapat.
- Memanggil method `tampilscore()` untuk menampilkan informasi skor.

5. tampilscore Method:

- untuk menampilkan jumlah skor yang diperoleh.

The screenshot shows the Greenfoot software interface with the title bar "Latar - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". Below the menu is a toolbar with buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu "Source Code" is open. The main window displays Java code for the "Latar" class:

```
private void countTime()
{
    time--;
    showTime();
    if (time == 0)
    {
        backgroundMusic.stop();
        Greenfoot.stop();
        addObject(new Menang(), 250, 300);
    }
}

private void showTime()
{
    showText("Waktu :" + time, 250, 25);
}

public void act()
{
    if (Greenfoot.getRandomNumber (100) < 3)
    {
        addObject(new Musuh(), Greenfoot.getRandomNumber(599), 500);
    }
    countTime();
}

/**
```

6. countTime Method:

- Mengurangi waktu saat permainan dimulai
- Serta Menampilkan sisa waktu pada posisi tertentu di layar.
- Jika waktu habis itu juga akan menghentikan musik latar, menghentikan permainan, dan menambahkan objek 'Menang'.

7. showTime Method:

- Menampilkan sisa waktu pada posisi tertentu di layar.

8. Metode `act` ini memiliki dua fungsi utama:

- Mengurangi waktu dengan memanggil metode `countTime()`.
- Menambahkan objek Musuh secara acak jika hasil pemilihan acak kurang dari 3.

Dengan kata lain, setiap kali objek ini "beraksi" dalam permainan, waktu akan berkurang, dan ada peluang kecil bahwa objek Musuh baru akan ditambahkan ke dunia permainan.

Latar - CAR WARS

Class Edit Tools Options

CharSelect Latar

Compile Undo Cut Copy Paste Find... Close Source Code

```
addObject(new Menang(), 250, 300);
}

private void showTime()
{
    showText("Waktu :" + time, 250, 25);
}

public void act()
{
    if (Greenfoot.getRandomNumber (100) < 3)
    {
        addObject(new Musuh(), Greenfoot.getRandomNumber(599), 500);
    }
    countTime();
}

/**
 * Prepare the world for the start of the program.
 * That is: create the initial objects and add them to the world.
 */
private void prepare()
{

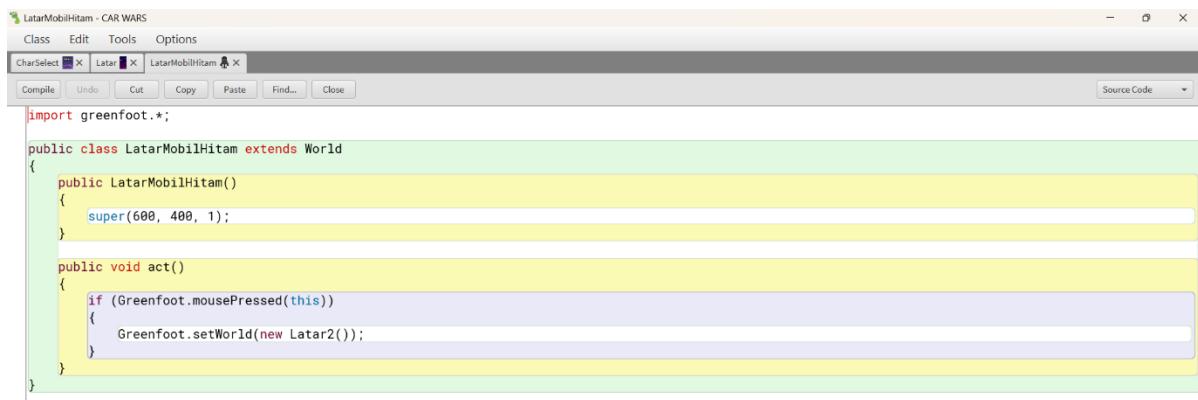
}
```

1 31°C Sebagian cerah Cari 13.33 23/12/2023

9. prepare Method:

Kelas ini bertanggung jawab atas logika dan tampilan utama permainan, termasuk manajemen nyawa, skor, waktu, dan musik latar.

LatarMobilHitam class



```
import greenfoot.*;  
public class LatarMobilHitam extends World  
{  
    public LatarMobilHitam()  
    {  
        super(600, 400, 1);  
    }  
    public void act()  
    {  
        if (Greenfoot.mousePressed(this))  
        {  
            Greenfoot.setWorld(new Latar2());  
        }  
    }  
}
```



1. Constructor (`public LatarMobilHitam()`):

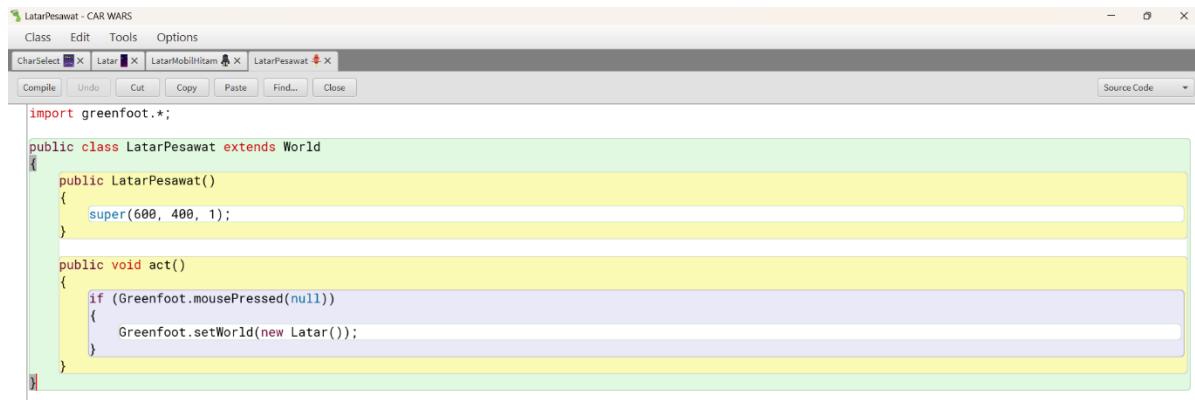
- berfungsi untuk membuat tampilan baru

2. act Method:

Saat memulai game latar akan berpindah ke latar2

Dengan kata lain, jika pemain menekan tombol mouse pada objek ini, permainan akan memindahkan mereka ke tingkat atau dunia selanjutnya, yaitu Latar2.

LatarPesawat



```
import greenfoot.*;  
  
public class LatarPesawat extends World  
{  
    public LatarPesawat()  
    {  
        super(600, 400, 1);  
    }  
  
    public void act()  
    {  
        if (Greenfoot.mousePressed(null))  
        {  
            Greenfoot.setWorld(new Latar());  
        }  
    }  
}
```



1. Constructor (public LatarPesawat()):
 - Berfungsi membuat tampilan baru
2. act Method:

Jika menekan tombol mouse, maka permainan akan memindahkan game ke dunia baru dari kelas Latar menggunakan Greenfoot.setWorld(new Latar()).

Kelas ini berfungsi sebagai dunia atau latar belakang yang memungkinkan perpindahan ke dunia berikutnya (Latar) ketika tombol mouse ditekan.

Menu

The screenshot shows a game development interface with a menu bar (Class, Edit, Tools, Options) and tabs (CharSelect, Latar, LatarMobilHitam, LatarPesawat, Menu). The main window displays the source code for the `Menu` class:

```
Menu - CAR WARS
Class Edit Tools Options
CharSelect X Latar X LatarMobilHitam X LatarPesawat X Menu X
Compile Undo Cut Copy Paste Find... Close Source Code
/*
 * Write a description of class Menu here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Menu extends World
{
    /**
     * Constructor for objects of class Menu.
     *
     */
    public Menu()
    {
        // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
        super(500, 600, 1);
        if(this.getClass().getName().equalsIgnoreCase("MenuUtama"))
        {
            TombolBantuan bantuan = new TombolBantuan();
            addObject(bantuan,361,510);
            TombolMulai mulai = new TombolMulai();
            addObject(mulai,141,510);
        }
        else
        {
            addObject (new TombolKembali(),61,61);
        }
    }
}
```

The code implements a constructor that creates a world of 600x400 cells. If the class name is "MenuUtama", it adds a "TombolBantuan" object at coordinates (361, 510) and a "TombolMulai" object at (141, 510). Otherwise, it adds a "TombolKembali" object at (61, 61). The code is annotated with Javadoc-style comments.

1. Constructor (public Menu()):

- Memeriksa apakah kelas saat ini adalah `MenuUtama`. Jika ya, maka menambahkan tombol bantuan dan tombol mulai ke dunia. Jika tidak, menambahkan tombol kembali ke dunia.

The screenshot shows the same game development interface with the source code for the `Menu` class. A new `private void prepare()` method has been added to the class:

```
Menu - CAR WARS
Class Edit Tools Options
CharSelect X Latar X LatarMobilHitam X LatarPesawat X Menu X
Compile Undo Cut Copy Paste Find... Close Source Code
/*
 * Constructor for objects of class Menu.
 *
 */
public Menu()
{
    // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
    super(500, 600, 1);
    if(this.getClass().getName().equalsIgnoreCase("MenuUtama"))
    {
        TombolBantuan bantuan = new TombolBantuan();
        addObject(bantuan,361,510);
        TombolMulai mulai = new TombolMulai();
        addObject(mulai,141,510);
    }
    else
    {
        addObject (new TombolKembali(),61,61);
    }
}

/**
 * Prepare the world for the start of the program.
 * That is: create the initial objects and add them to the world.
 */
private void prepare()
{
}
```

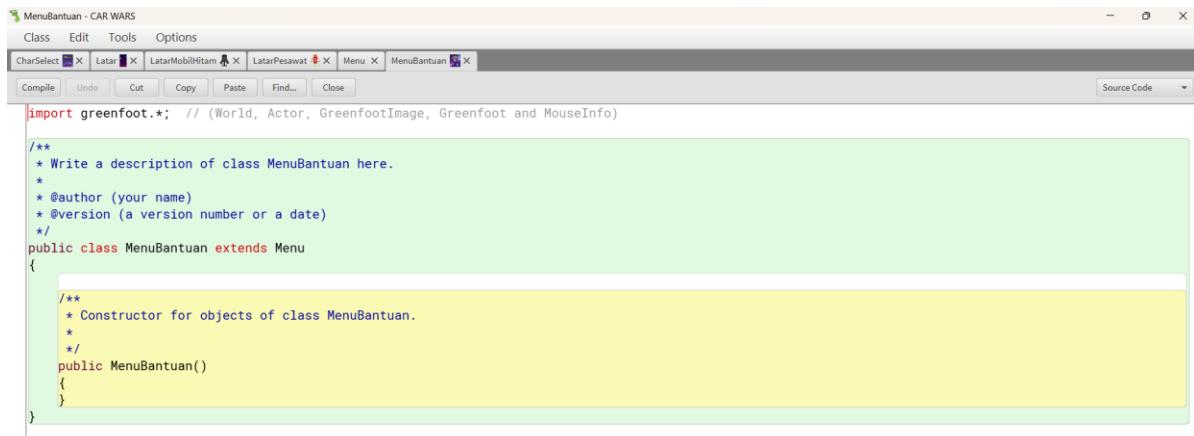
The `prepare()` method is intended to initialize the world by creating initial objects and adding them to the world.

2. prepare Method:

- Persiapkan game untuk memulai program

Kelas ini berfungsi sebagai latar belakang yang menampilkan tombol-tombol tertentu tergantung pada jenis kelas

MenuBantuan



The screenshot shows a Java code editor window titled "MenuBantuan - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". Below the menu bar is a toolbar with buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu "Source Code" is open. The code editor displays the following Java code:

```
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

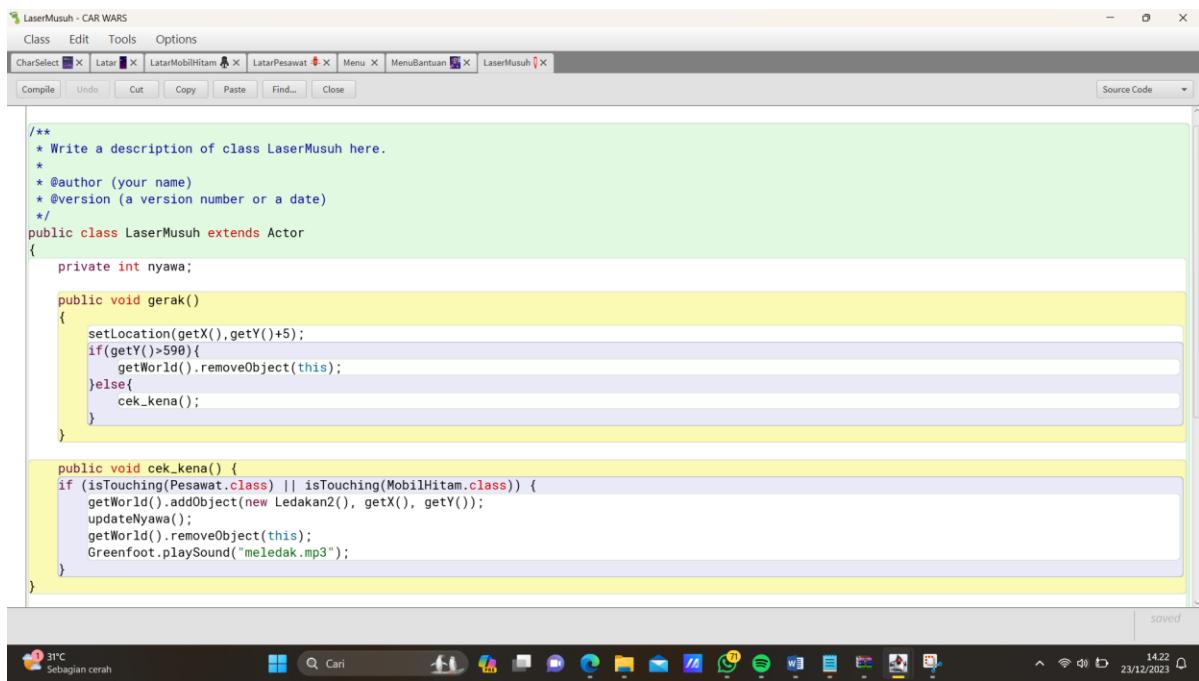
/**
 * Write a description of class MenuBantuan here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class MenuBantuan extends Menu
{
    /**
     * Constructor for objects of class MenuBantuan.
     */
    public MenuBantuan()
    {
    }
}
```



1. Constructor (public MenuBantuan()):

Kelas MenuBantuan merupakan subkelas dari Menu, yang berarti bahwa ia mewarisi properti dan perilaku dari kelas Menu tanpa mengubah apapun.

LaserMusuh



The screenshot shows the Greenfoot IDE interface with the project 'LaserMusuh - CAR WARS' open. The code editor displays the 'LaserMusuh' class, which extends 'Actor'. The class contains three methods: 'gerak()', 'cek_kena()', and a constructor. The 'gerak()' method moves the actor down by 5 units and removes it from the world if it reaches the bottom edge. The 'cek_kena()' method checks for collisions with 'Pesawat' or 'MobilHitam' objects and handles an explosion sound effect if a collision occurs.

```
/*
 * Write a description of class LaserMusuh here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class LaserMusuh extends Actor
{
    private int nyawa;

    public void gerak()
    {
        setLocation(getX(), getY() + 5);
        if (getY() > 590) {
            getWorld().removeObject(this);
        } else {
            cek_kena();
        }
    }

    public void cek_kena() {
        if (isTouching(Pesawat.class) || isTouching(MobilHitam.class)) {
            getWorld().addObject(new Ledakan2(), getX(), getY());
            updateNyawa();
            getWorld().removeObject(this);
            Greenfoot.playSound("meledak.mp3");
        }
    }
}
```

1. Variabel (private int nyawa;):
 - Variabel nyawa digunakan untuk menyimpan nilai nyawa.
2. Method gerak():
 - Method ini menggerakkan objek LaserMusuh ke bawah.
 - Jika posisi Y melebihi batas tertentu (590), objek LaserMusuh dihapus dari game.
 - Memanggil method cek_kena() untuk mengecek apakah objek LaserMusuh menyentuh Pesawat atau MobilHitam.
3. Method cek_kena():
 - Method ini mengecek apakah objek LaserMusuh menyentuh Pesawat atau MobilHitam.
 - Jika iya, menambahkan objek Ledakan2 ke dunia, memanggil method updateNyawa(), dan menghapus objek LaserMusuh method ini juga untuk memutar suara ledakan saat laser saat mengenai objek.

```

LaserMusuh - CAR WARS
Class Edit Tools Options
CharSelect X Latar X LatarMobilHitam X LatarPesawat X Menu X MenuBantuan X LaserMusuh X
Compile Undo Cut Copy Paste Find... Close Source Code
}

public void cek_kena() {
    if (isTouching(Pesawat.class) || isTouching(MobilHitam.class)) {
        getWorld().addObject(new Ledakan2(), getX(), getY());
        updateNyawa();
        getWorld().removeObject(this);
        Greenfoot.playSound("meledak.mp3");
    }
}

private void updateNyawa() {
    World world = getWorld();
    if (world instanceof Latar) {
        Latar latar = (Latar) world;
        latar.updatenyawa(-1);
    } else if (world instanceof Latar2) {
        Latar2 latar2 = (Latar2) world;
        latar2.updatenyawa(-1);
    }
}

public void act()
{
    gerak();
}
}

```

31°C Sebagian cerah 14.22 23/12/2023

4. Method updateNyawa():

- Method ini berfungsi untuk memberi informasi berapa nyawa yang masih kita miliki

5. Method act():

- Method ini memiliki fungsi memanggil metode gerak()ndisetiap frame game
-

Ledakan

```

Ledakan1 - CAR WARS
Class Edit Tools Options
CharSelect X Latar X LatarMobilHitam X LatarPesawat X Menu X MenuBantuan X LaserMusuh X Ledakan1 X
Compile Undo Cut Copy Paste Find... Close Source Code
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/**
 * Write a description of class Ledakan1 here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Ledakan1 extends Actor
{
    private int jeda=5;
    /**
     * Act - do whatever the Ledakan1 wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void act()
    {
        if(jeda>0)jeda--;
        else getWorld().removeObject(this);
    }
}

```

USD/IDR -0,26% 14.25 23/12/2023

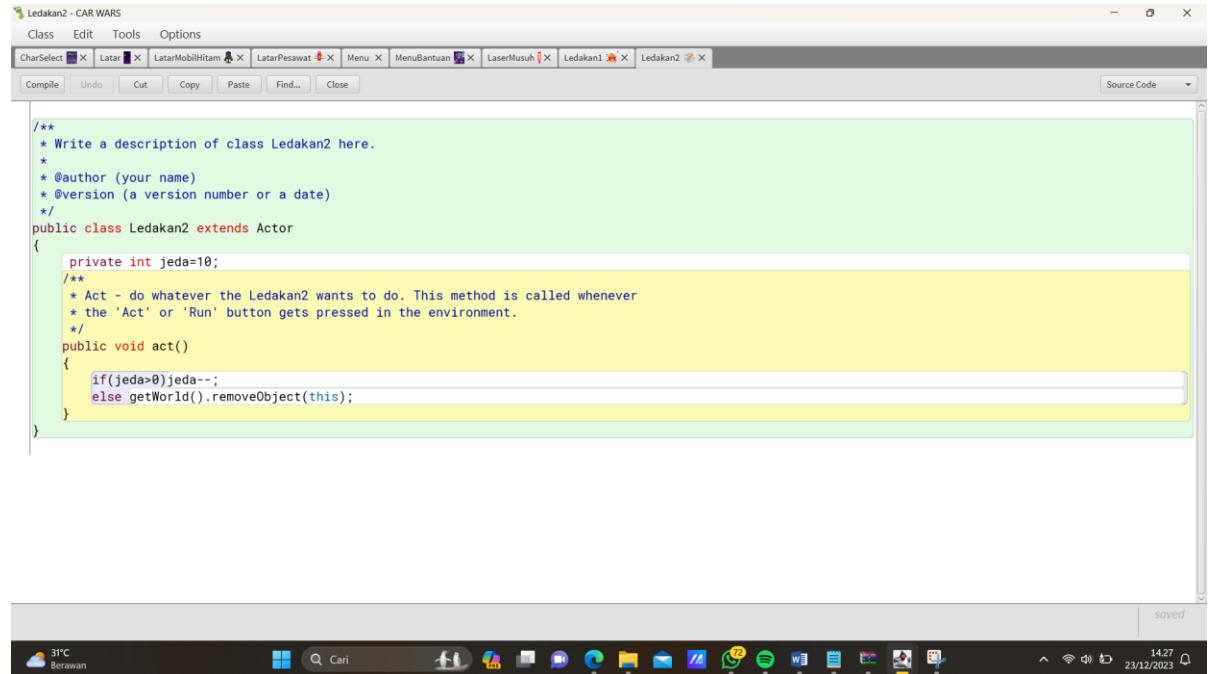
1. Variabel (private int jeda = 5;):

- Variabel jeda ini digunakan sebagai penghitung waktu, saat ledakan1 muncul, ledakan itu akan tetap ada dalam waktu yang singkat hingga akhirnya terhapus

2. Method act():

Berfungsi untuk mengecek jeda ledakan1 dalam waktu yang sudah ditentukan, jika ledakan1 sudah mencapai waktu yang ditentukan maka akan dihapuskan dari frame game

Ledakan2



```
/**  
 * Write a description of class Ledakan2 here.  
 *  
 * @author (your name)  
 * @version (a version number or a date)  
 */  
public class Ledakan2 extends Actor  
{  
    private int jeda=10;  
    /**  
     * Act - do whatever the Ledakan2 wants to do. This method is called whenever  
     * the 'Act' or 'Run' button gets pressed in the environment.  
     */  
    public void act()  
    {  
        if(jeda>0)jeda--;  
        else getWorld().removeObject(this);  
    }  
}
```

Sama seperti penjelasan ledakan yang membedakan hanya batas waktu jedanya

Menang

The screenshot shows the Greenfoot IDE interface with the title bar "Menang - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". A toolbar below the menu has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu "Source Code" is open. The main code editor contains the following Java code:

```
/** * Write a description of class Menang here. * * @author (your name) * @version (a version number or a date) */ public class Menang extends Actor {    /**     * Act - do whatever the Menang wants to do. This method is called whenever     * the 'Act' or 'Run' button gets pressed in the environment.     */    public void act()    {        // Add your action code here.    } }
```

The status bar at the bottom shows "saved" and system icons.

1. Method act():

Methode ini digunakan untuk memberikan aksi saat objek menang

MobilHitam

The screenshot shows the Greenfoot IDE interface with the title bar "MobilHitam - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". A toolbar below the menu has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu "Source Code" is open. The main code editor contains the following Java code:

```
public class MobilHitam extends Actor {    public int tahan=10;    private int jeda=0;    private int nyawa;    public void tombol(){        if(Greenfoot.isKeyDown("up")){            setLocation(getX(), getY()-4);        }        if(Greenfoot.isKeyDown("down")){            setLocation(getX(), getY()+4);        }        if(Greenfoot.isKeyDown("left")){            setLocation(getX()-4, getY());        }        if(Greenfoot.isKeyDown("right")){            setLocation(getX()+4, getY());        }        if (tahan==0){            if(Greenfoot.isKeyDown("Space"))            {                getWorld().addObject(new Rudal12(), getX(), getY());                Greenfoot.playSound("bullet.wav");            }            tahan=10;        }    } }
```

The status bar at the bottom shows "Berawan" and system icons.

1. Variabel:

- tahan: berfungsi untuk memberikan jeda antara tembakan, lalu nilai akan berkurang setiap tembakan mengenai objek.

2. Method tombol():

- Berfungsi untuk mengemudikan objek Mobil Hitam ke arah atas, bawah, kiri, dan kanan, tergantung dari tombol yang kita gunakan, lalu untuk mengeluarkan tembakan atau serangan menggunakan tombol space

The screenshot shows the Greenfoot software interface with the title bar "MobilHitam - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". The main window displays the Java source code for the MobilHitam class. The code handles key input for movement and shooting. It includes logic for moving the car up, down, left, and right based on keyboard input. It also checks if the space bar is pressed to shoot a bullet. A variable 'tahan' is used to manage the bullet delay. The code is color-coded with syntax highlighting for keywords like 'if', 'public', 'void', and 'else'. The status bar at the bottom shows weather information ("31°C Berawan"), system icons, and the date/time ("23/12/2023 14:31").

```
if(Greenfoot.isKeyDown("down")){
    setLocation(getX(), getY() + 4);
}
if(Greenfoot.isKeyDown("left")){
    setLocation(getX() - 4, getY());
}
if(Greenfoot.isKeyDown("right")){
    setLocation(getX() + 4, getY());
}
if (tahan == 0){
    if(Greenfoot.isKeyDown("Space")){
        getWorld().addObject(new Rudal2(), getX(), getY());
        Greenfoot.playSound("bullet.wav");
    }
    tahan = 10;
}
}

public void act()
{
    tahan--;
    tombol();
    if(jeda > 0) jeda--;
    if(jeda == 1)
        if(jeda == 0) jeda = 20;
}
```

3. Method act():

- Method ini dipanggil setiap layar di refresh, lalu didalamnya memanggil method tombol() untuk mengatur gerakan mobil dan serangannya

Musuh

The screenshot shows the Greenfoot IDE interface with the title bar "Musuh - CAR WARS". The menu bar includes Class, Edit, Tools, and Options. The toolbar has buttons for Compile, Undo, Cut, Copy, Paste, Find..., and Close. A dropdown menu "Source Code" is open. The code editor contains the following Java code:

```
* @author (your name)
* @version (a version number or a date)
*/
public class Musuh extends Actor
{
    public int tahan=30;
    private int jeda=0;

    public void gerak()
    {
        setLocation(getX(), getY() + 1);
        if (getY() > 500) {
            setLocation(Greenfoot.getRandomNumber(500),
                        Greenfoot.getRandomNumber(50));
        }
    }

    public void act()
    {
        gerak();
        if (tahan == 0) {
            tahan = 30;
        }
        if (jeda > 0) jeda--;
        if (jeda == 1) getWorld().addObject(new LaserMusuh(), getX(), getY() + 50);
        if (jeda == 0) jeda = 120;
    }
}
```

The status bar at the bottom shows "saved" and system icons like battery level (37%), network, and date/time (23/12/2023 14:35).

1. Variabel:

- tahan: Digunakan untuk memberikan penundaan antara pergerakan Musuh. Nilai ini akan berkurang setiap act cycle.
- jeda: Saat nol, Musuh akan menembakkan laser. Setelah itu, diatur kembali ke 120 untuk menunggu tembakan berikutnya.

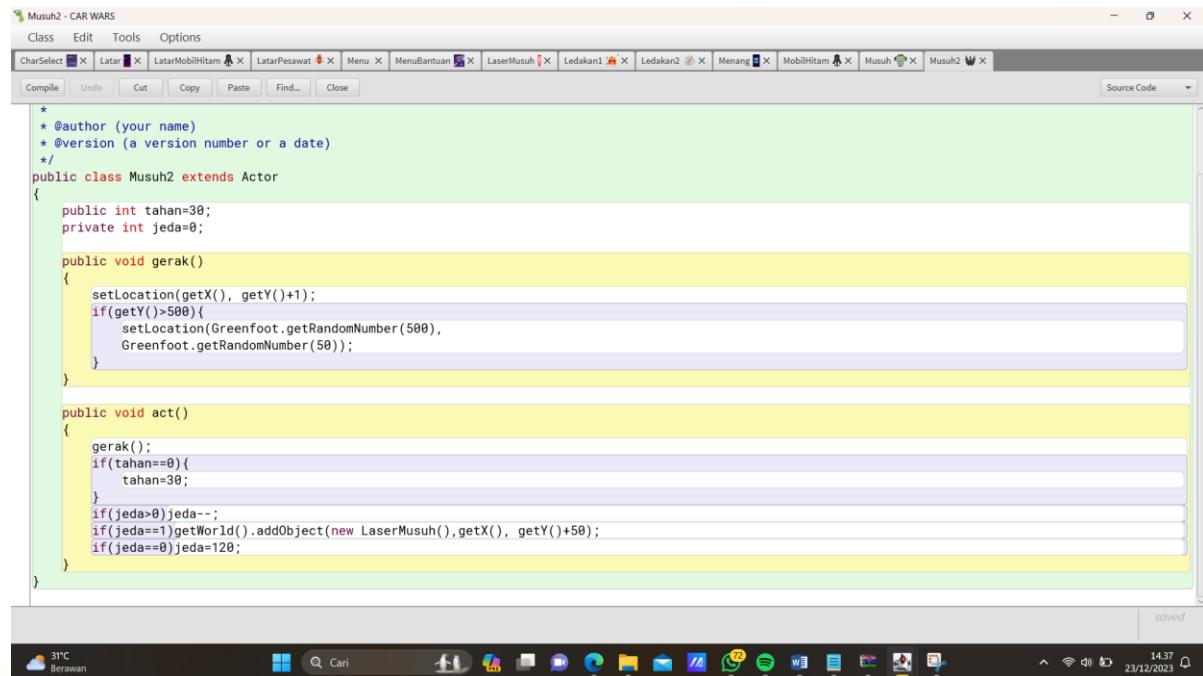
2. Method gerak():

- Digunakan untuk menggerakkan Musuh ke bawah layar.
- Jika Musuh mencapai batas bawah layar, posisinya direset ke atas layar dengan posisi acak.

3. Method act():

- Dipanggil setiap frame.
- Memanggil method gerak().
- Mengurangkan nilai tahan.
- Menangani tembakan laser saat jeda mencapai 1. Jeda tembakan diatur ulang ke 120 untuk menunggu tembakan berikutnya.

Musuh2



The screenshot shows the Greenfoot IDE interface with the title bar "Musuh2 - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu "Source Code" is open. The code editor contains the following Java code:

```
* @author (your name)
* @version (a version number or a date)
*/
public class Musuh2 extends Actor
{
    public int tahan=30;
    private int jeda=0;

    public void gerak()
    {
        setLocation(getX(), getY() + 1);
        if (getY() > 500) {
            setLocation(Greenfoot.getRandomNumber(500),
                        Greenfoot.getRandomNumber(50));
        }
    }

    public void act()
    {
        gerak();
        if (tahan == 0) {
            tahan = 30;
        }
        if (jeda > 0) jeda--;
        if (jeda == 1) getWorld().addObject(new LaserMusuh(), getX(), getY() + 50);
        if (jeda == 0) jeda = 120;
    }
}
```

The status bar at the bottom shows "saved", the system tray with icons for battery, signal, and date/time (23/12/2023), and the taskbar with various application icons.

1. Variabel:

- tahan: Digunakan untuk memberikan penundaan antara pergerakan Musuh2. Nilai ini akan berkurang setiap act cycle.
- jeda: Saat nol, Musuh2 akan menembakkan laser. Setelah itu, diatur kembali ke 120 untuk menunggu tembakannya.

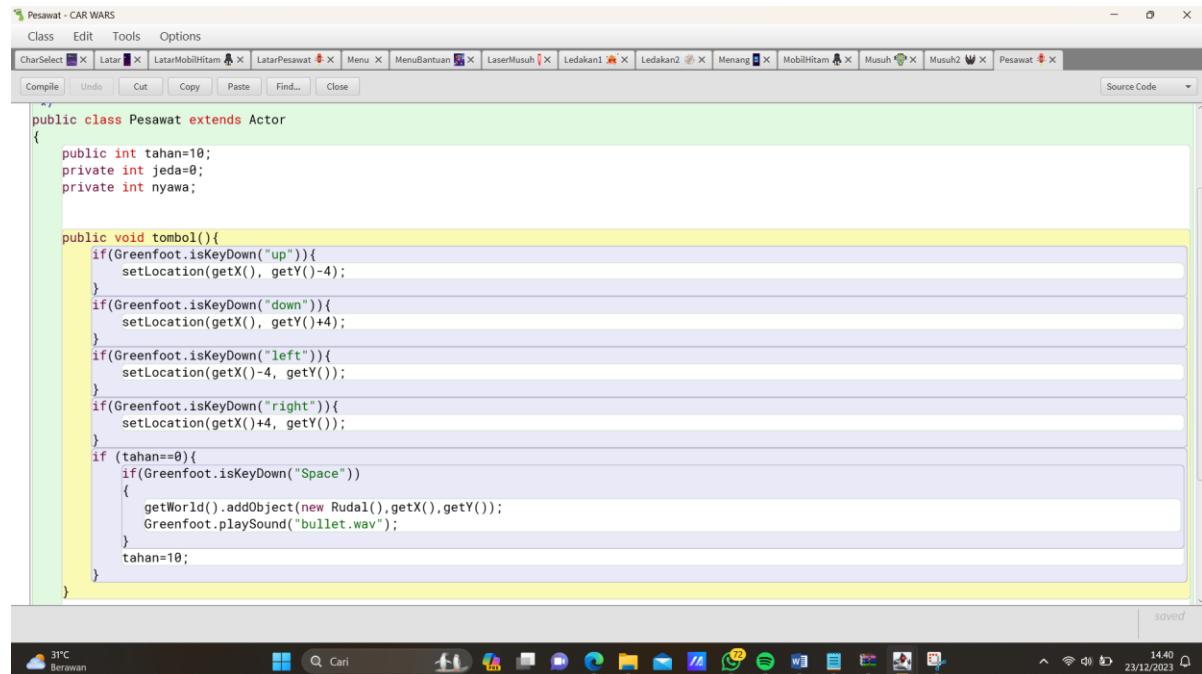
2. Method gerak():

- Digunakan untuk menggerakkan Musuh2 ke bawah layar.
- Jika Musuh2 mencapai batas bawah layar, posisinya direset ke atas layar dengan posisi acak.

3. Method act():

- Dipanggil setiap frame.
- Memanggil method gerak().
- Mengurangkan nilai tahan.
- Menangani tembakan laser saat jeda mencapai 1. Jeda tembakan diatur ulang ke 120 untuk menunggu tembakannya.

Pesawat



The screenshot shows the Greenfoot IDE interface with the title "Pesawat - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A tab bar at the top lists various classes: CharSelect, Latar, LatarMobilHitam, LatarPesawat, Menu, MenuBantuan, LaserMusuh, Ledakan1, Ledakan2, Menang, MobilHitam, Musuh, and Musuh2. The main window displays the source code for the "Pesawat" class, which extends "Actor". The code implements movement logic based on key presses and handles a bullet firing event.

```
public class Pesawat extends Actor
{
    public int tahan=10;
    private int jeda=0;
    private int nyawa;

    public void tombol(){
        if(Greenfoot.isKeyDown("up")){
            setLocation(getX(), getY()-4);
        }
        if(Greenfoot.isKeyDown("down")){
            setLocation(getX(), getY()+4);
        }
        if(Greenfoot.isKeyDown("left")){
            setLocation(getX()-4, getY());
        }
        if(Greenfoot.isKeyDown("right")){
            setLocation(getX()+4, getY());
        }
        if (tahan==0){
            if(Greenfoot.isKeyDown("Space")){
                {
                    getWorld().addObject(new Rudal(), getX(), getY());
                    Greenfoot.playSound("bullet.wav");
                }
                tahan=10;
            }
        }
    }
}
```

1. Variabel:

- tahan: Digunakan untuk memberikan penundaan antara tembakan Pesawat. Nilai ini akan berkurang setiap act cycle.
- jeda: Setelah setiap tembakan, nilai ini digunakan untuk memberikan jeda sebelum Pesawat dapat menembak lagi.

2. Method tombol():

- Digunakan untuk mengontrol gerakan dan tembakan Pesawat.
- Menggunakan kunci panah untuk menggerakkan Pesawat ke atas, ke bawah, ke kiri, dan ke kanan.
- Menggunakan kunci "Space" untuk menembakkan rudal.

The screenshot shows the Greenfoot IDE interface with the title bar "Pesawat - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A tab bar at the top lists various classes: CharSelect, Latar, LatarMobilHitam, LatarPesawat, Menu, MenuBantuan, LaserMusuh, Ledakan1, Ledakan2, Menang, MobilHitam, Musuh, Musuh2, and Pesawat. A "Source Code" dropdown is open. The main code editor contains the following Java code:

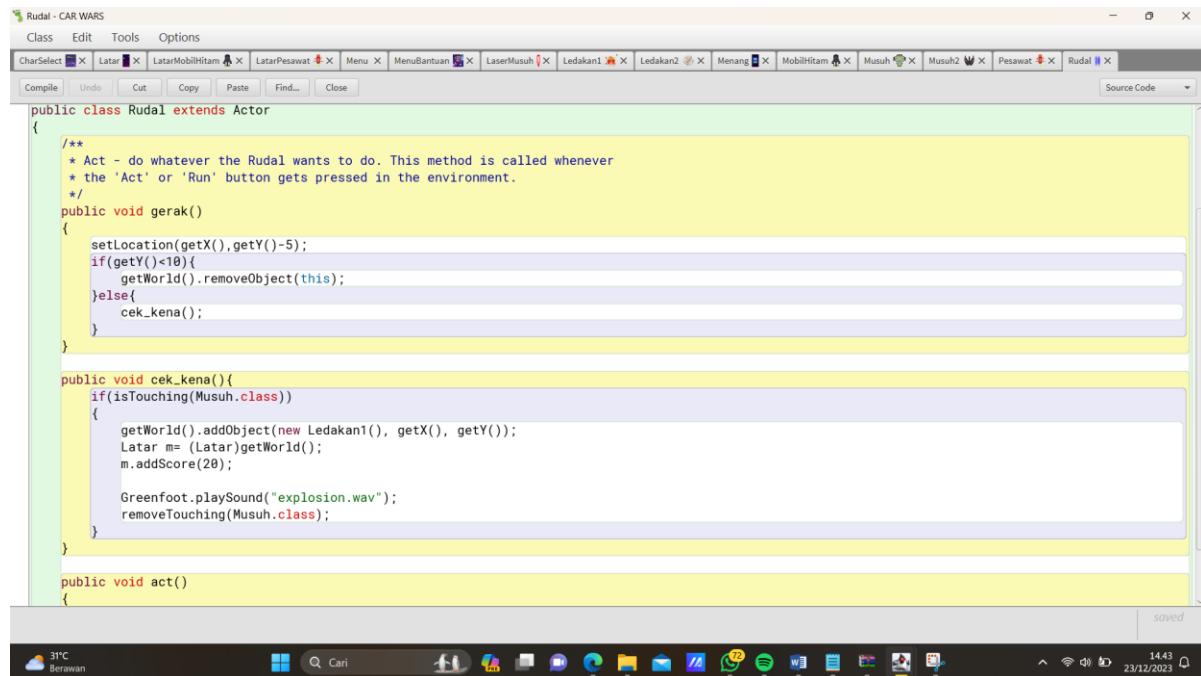
```
if(Greenfoot.isKeyDown("down")){
    setLocation(getX(), getY()+4);
}
if(Greenfoot.isKeyDown("left")){
    setLocation(getX()-4, getY());
}
if(Greenfoot.isKeyDown("right")){
    setLocation(getX()+4, getY());
}
if (tahan==0){
    if(Greenfoot.isKeyDown("Space"))
        getWorld().addObject(new Rudal(), getX(), getY());
    Greenfoot.playSound("bullet.wav");
    tahan=10;
}
public void act()
{
    tahan--;
    tombol();
    if(jeda>0)jeda--;
    if(jeda==1)
        if(jeda==0)jeda=20;
}
```

The status bar at the bottom shows "saved", weather "Berawan", temperature "31°C", and system information including battery level, signal strength, and date/time "23/12/2023 14:40".

3. Method act():

- Dipanggil setiap frame.
- Memanggil method tombol() untuk mengontrol gerakan dan tembakan Pesawat.
- Mengurangkan nilai tahan.
- Mengatur jeda tembakan.

Rudal



The screenshot shows the Greenfoot IDE interface with the Rudal class selected. The code implements movement logic for the Rudal actor, checks for collisions with enemies, and handles explosions.

```
public class Rudal extends Actor {
    /**
     * Act - do whatever the Rudal wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void gerak() {
        setLocation(getX(), getY() - 5);
        if (getY() < 10) {
            getWorld().removeObject(this);
        } else {
            cek_kena();
        }
    }

    public void cek_kena() {
        if (isTouching(Musuh.class)) {
            getWorld().addObject(new Ledakan1(), getX(), getY());
            Latar m = (Latar) getWorld();
            m.addScore(20);

            Greenfoot.playSound("explosion.wav");
            removeTouching(Musuh.class);
        }
    }

    public void act()
    {
    }
}
```

1. Method gerak():

- Digunakan untuk mengatur gerakan rudal.
- Mengurangkan posisi Y rudal sehingga tampak bergerak ke atas.
- Jika rudal mencapai batas atas layar, maka rudal dihapus.

2. Method cek_kena():

- Digunakan untuk mengecek tabrakan rudal dengan musuh.
- Jika rudal menyentuh musuh, efek ledakan (Ledakan1) ditambahkan di posisi rudal.
- Poin skor ditambahkan menggunakan method addScore pada kelas Latar.
- Suara ledakan (explosion.wav) dimainkan.
- Objek musuh yang disentuh dihapus.

Rudal2

The screenshot shows the Greenfoot IDE interface with the title bar "Rudal2 - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A "Source Code" dropdown is also present. The code editor contains the following Java code:

```
public class Rudal2 extends Actor
{
    /**
     * Act - do whatever the Rudal2 wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void gerak2()
    {
        setLocation(getX(), getY() - 5);
        if(getY() < 10)
        {
            getWorld().removeObject(this);
        }
        else{
            cek_kena2();
        }
    }

    public void cek_kena2(){
        if(isTouching(Musuh.class))
        {
            getWorld().addObject(new Ledakan1(), getX(), getY());
            Latar2 m= (Latar2) getWorld();
            m.addScore(20);

            Greenfoot.playSound("explosion.wav");
            removeTouching(Musuh.class);
        }
    }

    public void act()
    {
    }
}
```

Below the code editor, a status bar displays "Class compiled - no syntax errors" and "saved". The system tray at the bottom shows icons for weather (31°C Berawan), search (Cari), and various system functions.

TombolBantuan

The screenshot shows the Greenfoot IDE interface with the title bar "TombolBantuan - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A "Source Code" dropdown is also present. The code editor contains the following Java code:

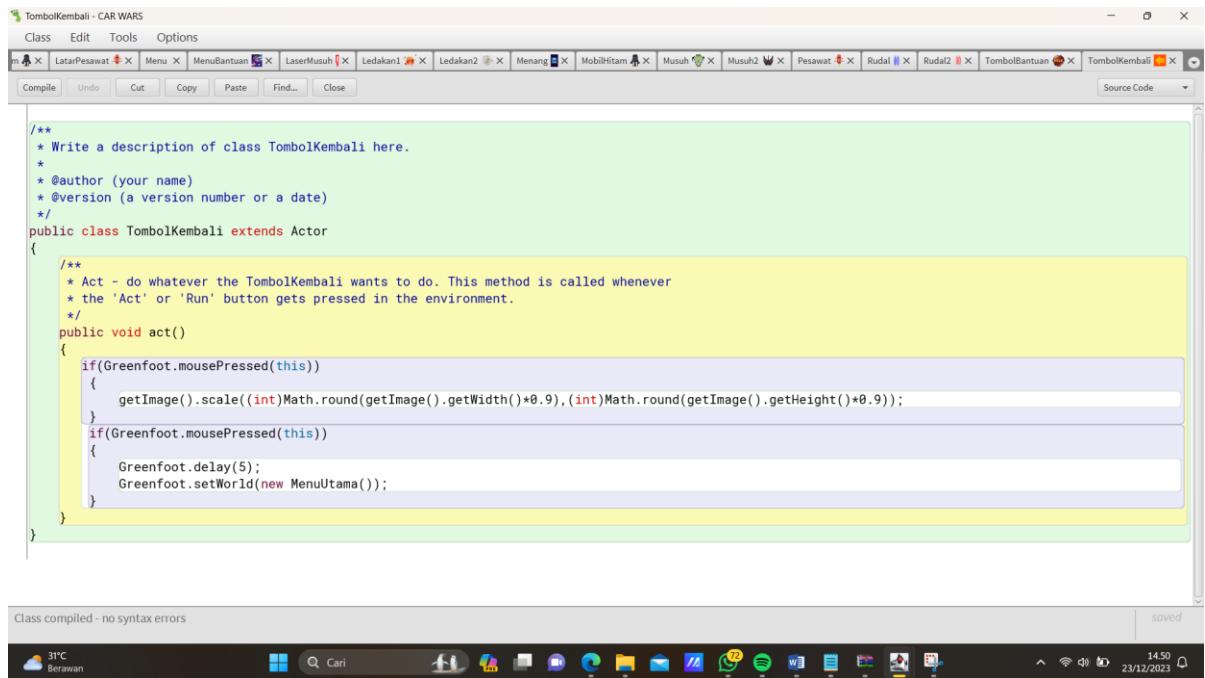
```
public class TombolBantuan extends Actor
{
    /**
     * Write a description of class TombolBantuan here.
     *
     * @author (your name)
     * @version (a version number or a date)
     */
    public void act()
    {
        if(Greenfoot.mousePressed(this))
        {
            getImage().scale((int) Math.round(getImage().getWidth() * 0.9), (int) Math.round(getImage().getHeight() * 0.9));
        }
        if(Greenfoot.mousePressed(this))
        {
            Greenfoot.delay(5);
            Greenfoot.setWorld(new MenuBantuan());
        }
    }
}
```

Below the code editor, a status bar displays "Class compiled - no syntax errors" and "saved". The system tray at the bottom shows icons for weather (31°C Berawan), search (Cari), and various system functions.

1. Method act():

- saat menekan tombol gambar akan otomatis mengecil 90% dari ukuran aslinya, lalu memberikan jeda sebelum game dimulai

TombolKembali



The screenshot shows the Greenfoot IDE interface with the title bar "TombolKembali - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A dropdown menu "Source Code" is open. The code editor contains the following Java code:

```
/**  
 * Write a description of class TombolKembali here.  
 *  
 * @author (your name)  
 * @version (a version number or a date)  
 */  
public class TombolKembali extends Actor  
{  
    /**  
     * Act - do whatever the TombolKembali wants to do. This method is called whenever  
     * the 'Act' or 'Run' button gets pressed in the environment.  
     */  
    public void act()  
    {  
        if(Greenfoot.mousePressed(this))  
        {  
            getImage().scale((int)Math.round(getImage().getWidth()*0.9),(int)Math.round(getImage().getHeight()*0.9));  
        }  
        if(Greenfoot.mousePressed(this))  
        {  
            Greenfoot.delay(5);  
            Greenfoot.setWorld(new MenuUtama());  
        }  
    }  
}
```

The status bar at the bottom left says "Class compiled - no syntax errors" and "saved". The taskbar at the bottom shows various application icons.

Sama seperti TombolBantuan yang membedakan hanya TombolKembali membantu ke halaman MenuUtama

TombolMulai

The screenshot shows the Greenfoot IDE interface with the window titled "TombolMulai - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar below has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A "Source Code" dropdown is also present. The code editor contains the following Java code:

```
/** * Write a description of class TombolMulai here. * * @author (your name) * @version (a version number or a date) */ public class TombolMulai extends Actor {    /**     * Act - do whatever the TombolMulai wants to do. This method is called whenever     * the 'Act' or 'Run' button gets pressed in the environment.     */    public void act()    {        if(Greenfoot.mousePressed(this))        {            getImage().scale((int)Math.round(getImage().getWidth()*0.9),(int)Math.round(getImage().getHeight()*0.9));            // Ganti kelas World yang diinstansiasi dengan CharSelect            Greenfoot.setWorld(new CharSelect());        }    } }
```

The status bar at the bottom indicates "Class compiled - no syntax errors" and shows the date "23/12/2023". The taskbar at the very bottom displays various application icons.

1. Method act():

- Untuk memunculkan tombol memulai game

TombolKalah

The screenshot shows the Greenfoot IDE interface with the window titled "kalah - CAR WARS". The menu bar includes "Class", "Edit", "Tools", and "Options". The toolbar below has buttons for "Compile", "Undo", "Cut", "Copy", "Paste", "Find...", and "Close". A "Source Code" dropdown is also present. The code editor contains the following Java code:

```
import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)  
/** * Write a description of class kalah here. * * @author (your name) * @version (a version number or a date) */ public class kalah extends Actor {    /**     * Act - do whatever the kalah wants to do. This method is called whenever     * the 'Act' or 'Run' button gets pressed in the environment.     */    public void act()    {        // Add your action code here.    } }
```

The status bar at the bottom indicates "Class compiled - no syntax errors" and shows the date "23/12/2023". The taskbar at the very bottom displays various application icons.

1. Method act():

- Dipanggil ketika kelas ini beraksi.
- Ini adalah tempat di mana kode tindakan akan ditambahkan sesuai dengan kebutuhan.

BAB III

KESIMPULAN

Dalam menutup pembahasan ini, dapat disarikan bahwa Greenfoot merupakan alat yang sangat efektif untuk memahami dan merancang pengembangan game, terutama bagi pemula. Dengan menerapkan konsep-konsep seperti aktor, world, dan skenario, Greenfoot memberikan dasar yang kokoh untuk menciptakan pengalaman bermain yang menarik.

Pemahaman terhadap pemrograman berbasis objek menjadi kunci untuk mengoptimalkan potensi penuh Greenfoot, sementara keberadaan editor yang mudah digunakan memudahkan proses pengembangan. Adanya fitur tambahan seperti animasi, suara, dan kemampuan untuk berkolaborasi juga memberikan daya tarik dan tingkat kompleksitas yang lebih tinggi pada game yang dihasilkan.

Keahlian dalam pemecahan masalah dan optimisasi dalam pengembangan game sangat penting. Greenfoot memberikan platform bagi pengembang untuk mengasah keterampilan teknis mereka sekaligus menggali kreativitas guna menciptakan pengalaman bermain yang menarik.

Sebagai hasilnya, dapat disimpulkan bahwa Greenfoot memiliki potensi besar bagi pengembang, baik yang baru memulai perjalanan maupun yang ingin lebih mendalam dalam dunia pengembangan game. Dengan berbagai fitur dan kemudahan yang disediakan, Greenfoot membuka peluang eksplorasi dan inovasi dalam menciptakan game yang tidak hanya menarik, tetapi juga mendidik.