

PRAKTIKUM PEMOGRAMAN WEB

Oleh Kelompok :

-

-

-

Teknik Informatika Sains & Teknologi

Universitas Muhammadiyah Kalimantan Timur

Samarinda, 2023

**Pendahuluan**

Tetris, salah satu game paling ikonik dalam sejarah video game, telah direalisasikan kembali menggunakan platform pembelajaran pemrograman Java yang disebut Greenfoot. Game ini memungkinkan pengguna untuk memahami konsep dasar pemrograman sambil menciptakan dan memodifikasi versi Tetris mereka sendiri. Laporan ini bertujuan untuk menyelidiki implementasi Tetris dalam Greenfoot, menguraikan tujuan proyek, metodologi pengembangan, serta rangkaian laporan untuk memahami struktur code dalam game ini.

**Tujuan Proyek**

1. Pengenalan Konsep Pemrograman: Memberikan pemahaman dasar tentang bahasa pemrograman Java kepada pengguna yang ingin mempelajari atau meningkatkan keterampilan mereka.
2. Pengembangan Kreativitas: Mendorong pengguna untuk bereksperimen dengan kode dan modifikasi untuk menghasilkan variasi permainan Tetris yang unik.
3. Pemahaman Interaktif: Mengajarkan prinsip-prinsip dasar interaksi antara objek dalam lingkungan permainan dengan menggunakan Greenfoot.

**Metodologi Pengembangan**

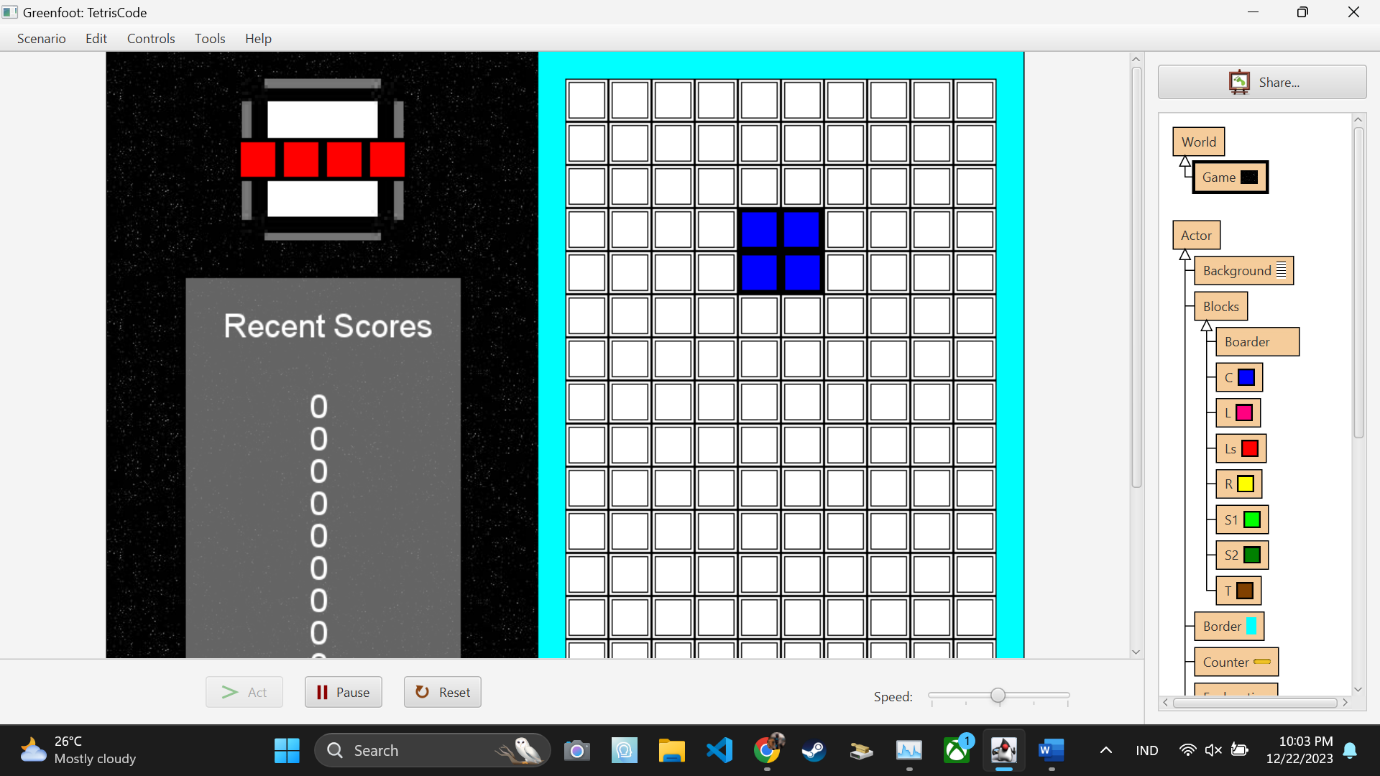
1. Pemahaman Konsep: Pemahaman terhadap aturan dan logika permainan Tetris.
2. Pembelajaran Greenfoot: Penggunaan dokumentasi dan sumber daya Greenfoot untuk mempelajari cara menggunakan platform ini.
3. Implementasi Dasar: Pembuatan struktur dasar permainan Tetris dalam Greenfoot.
4. Iterasi dan Penyesuaian: Meningkatkan gameplay, grafik, dan fitur tambahan dalam proyek.

**Rangkaian Laporan**

1. Pendahuluan: Pengenalan terhadap proyek, tujuan, dan metodologi pengembangan.
2. Deskripsi Code: Penjelasan struktur code untuk elemen kunci dalam permainan Tetris seperti blok, papan permainan, pergerakan, dan logika menghapus baris.
3. Modifikasi dan Peningkatan: Demonstrasi bagaimana code dapat dimodifikasi untuk menambahkan fitur baru atau mengubah mekanisme permainan.
4. Pengujian dan Evaluasi: Pengujian fungsionalitas game, identifikasi potensi perbaikan, dan evaluasi hasil.

**Penjelasan Code dalam Game**

Implementasi Tetris dalam Greenfoot memanfaatkan prinsip dasar pemrograman Java. Struktur code mencakup kelas-kelas yang mewakili setiap elemen permainan, seperti:



1. **Word**
   1. Game

Import Statements:

* + - Di bagian awal code, terdapat import statements yang meng-import kelas-kelas dari pustaka Greenfoot dan Java.util. Greenfoot adalah lingkungan pengembangan permainan sederhana yang menggunakan Java.

Variabel Kelas Game:

* + - Mendefinisikan variabel-variabel yang diperlukan untuk menjalankan permainan, seperti array gameboard, objek-objek Shape (blok-blok), counter untuk skor, variabel delayTime untuk mengatur kecepatan permainan, variabel level, dan objek GreenfootSound untuk mengontrol musik latar.

Constructor dan Metode Game:

* + - Constructor inisialisasi dunia permainan dan memanggil metode prepare().
    - act() adalah metode yang dipanggil setiap siklus permainan, yang dalam hal ini berisi pemanggilan moveThatBlock().

Metode Utama:

* + - makeIt() membuat blok baru dan mengecek apakah permainan sudah selesai.
    - initializeWorld() menginisialisasi dunia permainan dengan menambahkan elemen-elemen seperti background, border, counter skor, dan memulai musik latar.
    - moveThatBlock() mengatur gerakan blok utama berdasarkan input pemain dari keyboard.
    - convertIntoBlocks() mengubah data dari array gameboard ke representasi visual di layar.

Metode Tambahan:

* + - delays() mengatur delay dalam permainan berdasarkan tingkat dan skor.
    - lineCheck() mengecek jika ada baris yang terisi penuh untuk dihapus.
    - redraw() memperbarui tampilan permainan setelah ada baris yang terhapus.
    - gameOver() menampilkan layar Game Over ketika permainan selesai.
    - reset() mengatur ulang permainan ke kondisi awal.

Metode Spawner:

* + - spawner() dan firstBlockSpawn() memilih dan menampilkan blok-blok yang akan datang dan blok pertama saat permainan dimulai.

Metode untuk Interaksi Blok:

* + - Beberapa metode, seperti moveIntoPlay() dan convertIntoBlocks(), mengatur pergerakan, rotasi, dan visualisasi dari blok-blok yang muncul dan berinteraksi dalam permainan.

Kode ini mengontrol logika permainan Tetris, mulai dari mengatur blok-blok, menghapus baris, hingga memperbarui tampilan layar.

1. **Actor**
   1. Background

Import Statements:

* + - Seperti biasa, import statements digunakan untuk meng-import kelas-kelas dari pustaka Greenfoot yang dibutuhkan untuk mengembangkan permainan.

Kelas Background:

* + - Background merupakan subclass dari kelas Actor dalam Greenfoot. Kelas ini bertujuan untuk mengatur latar belakang atau background dalam permainan.

Metode act():

* + - Metode ini didefinisikan karena kelas ini adalah subclass dari Actor. Meskipun saat ini metode ini kosong (tanpa tindakan), biasanya metode ini digunakan untuk menentukan perilaku atau tindakan yang harus dilakukan oleh objek Background setiap kali siklus permainan berjalan.

Metode makeBackground():

* + - Meskipun metode ini ada dalam code, isinya kosong. Terlihat bahwa ada deklarasi variabel background yang mengambil gambar (image) dari objek Background sendiri, tetapi kode di dalamnya kosong sehingga tidak ada tindakan yang dijalankan dalam metode ini.
  1. Blocks

Import Statements:

* + - Mirip dengan sebelumnya, import statements digunakan untuk meng-import kelas-kelas dari pustaka Greenfoot yang dibutuhkan untuk pengembangan permainan.

Kelas Blocks:

* + - Blocks adalah sebuah kelas yang merupakan subclass dari Actor dalam Greenfoot. Kelas ini mungkin merupakan kerangka dasar untuk berbagai jenis blok atau elemen-elemen dalam permainan.

Metode act():

* + - Seperti halnya pada kelas Background, metode ini didefinisikan karena kelas Blocks adalah subclass dari Actor. Saat ini, metode ini kosong (tanpa tindakan). Biasanya, metode ini digunakan untuk menentukan perilaku atau tindakan yang harus dilakukan oleh objek Blocks setiap kali siklus permainan berjalan.
  1. Boarder

Import Statements:

* + - Seperti biasanya, ada import statements yang meng-import kelas-kelas dari pustaka Greenfoot yang dibutuhkan untuk pengembangan permainan.

Kelas Boarder:

* + - Boarder merupakan sebuah kelas yang meng-extends (memperluas) kelas Blocks. Dengan kata lain, Boarder mewarisi semua perilaku dan sifat-sifat dari kelas Blocks dan mungkin ditujukan untuk mewakili batas atau border dalam dunia permainan.

Metode act():

* + - Seperti pada kelas-kelas sebelumnya, metode act() didefinisikan di dalam kelas ini karena kelas Boarder adalah sebuah Actor dalam Greenfoot. Saat ini, metode ini kosong tanpa tindakan apapun yang dijalankan.
  1. Counter

Import Statements:

* Seperti biasa, ada import statements yang meng-import kelas-kelas dari pustaka Greenfoot yang diperlukan untuk pengembangan permainan.

Kelas Counter:

* Counter adalah sebuah kelas yang merupakan subclass dari Actor dalam Greenfoot. Kelas ini digunakan untuk menampilkan nilai numerik di layar permainan.

Variabel dan Metode:

* transparent: Mewakili warna transparan yang digunakan untuk latar belakang teks pada gambar.
* background: Gambar latar belakang untuk counter.
* value dan target: Menyimpan nilai aktual dan nilai target untuk counter.
* prefix: Digunakan untuk menampilkan teks sebelum nilai counter.
* updateImage(): Metode ini memperbarui gambar untuk menampilkan nilai saat ini di layar. Ini mencakup pembuatan gambar dengan teks nilai yang diperbarui.

Metode act():

* Metode ini dijalankan pada setiap siklus permainan. Ini menganimasikan peningkatan atau penurunan nilai counter secara bertahap menuju nilai target.

Metode addPoints():

* Metode ini menambahkan poin sesuai dengan jumlah garis yang dibersihkan dalam permainan Tetris. Ini memperbarui nilai counter dan mengembalikan jumlah poin yang sesuai.

Metode setValue(), getValue(), setPrefix(), add():

* Metode-metode ini digunakan untuk mengatur, mendapatkan, dan menambahkan nilai counter serta mengatur teks prefix.

Kelas Counter dapat digunakan untuk menampilkan berbagai jenis nilai di layar permainan, seperti skor, jumlah nyawa, atau elemen-elemen lain yang memerlukan representasi numerik. Codingan ini memberikan kemampuan untuk menyesuaikan tampilan nilai secara visual dengan menggunakan teks dan animasi sederhana.

* 1. Explanation

Variabel Kelas:

* width dan height: Mendefinisikan lebar dan tinggi gambar yang akan dibuat untuk menampilkan penjelasan.
* fontsize: Ukuran font teks yang akan digunakan.

Konstruktor Explanation():

* Konstruktor ini dipanggil saat membuat objek Explanation dan langsung memanggil metode window() untuk membuat tampilan penjelasan.

Metode window():

* Metode ini membuat gambar dengan latar belakang hitam transparan, dengan kotak putih semi-transparan di dalamnya.
* Mengatur font dan warna teks untuk menampilkan informasi tentang cara bermain permainan Tetris.
* Setelah menggambar teks, hasilnya diatur sebagai gambar untuk Actor.

Metode bugs():

* Metode ini mirip dengan window(), tetapi menampilkan informasi tentang "Bugs" dalam permainan.
* Ini juga membuat gambar dengan latar belakang hitam transparan dan kotak putih semi-transparan di dalamnya.
* Menampilkan pesan tentang tidak adanya bug yang ditemukan, meminta informasi tentang bug jika ada, dan memberikan penjelasan tentang lag dalam permainan.
* Seperti window(), hasil gambar diatur sebagai gambar untuk Actor.

Metode act():

* Metode ini kosong dan tidak melakukan apa pun. Biasanya, ini akan berisi logika yang akan dijalankan setiap siklus permainan.
* Kelas Explanation dirancang untuk menampilkan informasi dalam permainan, baik itu tentang cara bermain, pesan tertentu kepada pemain, atau informasi lain yang relevan dengan pengalaman permainan. Metode window() dan bugs() memungkinkan untuk menampilkan informasi yang berbeda berdasarkan pemanggilan metode tersebut.
  1. NextBlockBox

Kode ini adalah kelas NextBlockBox yang merupakan subclass dari Actor dalam Greenfoot. Kelas ini bertujuan untuk menampilkan kotak yang akan menampilkan blok berikutnya dalam permainan.

Namun, pada saat ini, kelas ini belum memiliki perilaku atau aksi yang diimplementasikan di dalam metode act(). Biasanya, metode act() akan digunakan untuk menentukan perilaku objek selama permainan, seperti pergerakan, interaksi dengan objek lain, atau perubahan status.

* 1. ScoreBoard

Metode ScoreBoard()

* Ini adalah konstruktor default yang tidak mengambil parameter apapun.

Metode ScoreBoard(String title,int score, int highscore)

* Konstruktor ini digunakan untuk membuat papan skor dengan judul tertentu (title). Metode ini menerima judul, nilai skor saat ini (score), dan skor tertinggi (highscore) sebagai argumen dan menampilkan informasi tersebut pada papan skor.

Metode ScoreBoard(int[] scores)

* Ini adalah konstruktor yang menerima sebuah array scores sebagai argumen. Metode ini digunakan untuk membuat papan skor dengan daftar skor terbaru. Setiap elemen array scores akan ditampilkan sebagai skor pada papan skor.

Metode makeCredits(boolean bol)

* Metode ini digunakan untuk menampilkan informasi kredit pada papan skor. Jika bol adalah false, maka informasi kredit akan ditampilkan. Jika bol adalah true, papan skor akan dihapus dari dunia permainan.

Metode makeScore(String header, String title, int score, int highscore)

* Metode ini bertujuan untuk membuat papan skor dengan informasi khusus. Parameter header adalah judul yang akan ditampilkan di papan skor, title adalah teks yang menyertai skor saat ini, score adalah nilai skor saat ini, dan highscore adalah skor tertinggi yang ditampilkan.

Metode makelist(int[] scores)

* Metode ini bertujuan untuk membuat papan skor dengan daftar skor terbaru. Ini menggunakan array scores untuk menampilkan daftar skor terbaru pada papan skor.
  1. Shape

Constructor dan Method act():

* Kelas ini memiliki method act() yang akan dijalankan setiap kali tombol 'Act' atau 'Run' ditekan dalam lingkungan permainan. Saat ini, act() tidak memiliki kode apapun.

Method rotate():

* Ini adalah method yang sepertinya akan bertanggung jawab untuk memutar bentuk (shape) dalam permainan. Namun, saat ini method ini kosong dan tidak melakukan apa pun.

Method createIt():

* Method ini mengembalikan sebuah array integer dengan satu elemen. Dalam kode saat ini, array tersebut hanya memiliki satu slot dan mengembalikan array tersebut.

Method checkDown(), checkLeft(), dan checkRight():

* Ketiga method ini mengembalikan nilai boolean (true dalam kode saat ini). Mereka mungkin bertanggung jawab untuk memeriksa apakah objek Shape dapat bergerak ke bawah, kiri, atau kanan dalam lingkungan permainan.
  1. Timer

Constructor dan Method act():

* Seperti kelas Shape sebelumnya, kelas ini juga memiliki method act() yang akan dijalankan setiap kali tombol 'Act' atau 'Run' ditekan dalam lingkungan permainan. Namun, pada saat ini, method act() tidak memiliki implementasi atau kode di dalamnya.

Kelas Timer ini mungkin bertanggung jawab untuk mengelola waktu atau menampilkan informasi waktu dalam permainan. Namun, karena method act() masih kosong, kelas ini belum melakukan apa pun secara fungsional. Potongan kode yang sebenarnya akan mengatur logika terkait waktu, penambahan/dikurangi waktu, atau penampilan waktu dalam permainan masih harus ditambahkan.

1. **Kesimpulan**

Dalam implementasi Tetris menggunakan Greenfoot, setiap kelas memiliki peran dan fungsinya sendiri dalam membangun permainan secara keseluruhan:

Game (World): Merupakan dunia permainan yang mengontrol semua elemen permainan seperti blok, skor, dan tampilan permainan. Metode-metodenya mengatur inisialisasi dunia, gerakan blok, pengecekan baris penuh, serta memperbarui tampilan saat permainan berlangsung.

Actor Subclasses (Background, Blocks, Border, Counter, Explanation, NextBlockBox, ScoreBoard, Shape, Timer): Setiap kelas ini mewakili elemen tertentu dalam permainan (misalnya, latar belakang, blok-blok, papan skor, penjelasan, dll.). Mereka memiliki metode act() yang biasanya mengatur perilaku objek dalam setiap siklus permainan.

Kelas Shape: Bertanggung jawab atas perilaku dan manipulasi blok-blok dalam permainan Tetris. Metode-metodenya diperkirakan mengatur pergerakan, rotasi, serta interaksi antar-blok.

Kelas Timer: Meskipun saat ini masih kosong, mungkin ditujukan untuk mengelola aspek waktu dalam permainan, seperti mengatur timer atau memperbarui tampilan waktu.

Tujuan dan Metodologi Pengembangan: Terdapat tujuan jelas dalam pengembangan proyek ini, seperti memberikan pemahaman dasar tentang Java, mendorong kreativitas, dan mengajarkan interaksi objek dalam permainan. Metodologi pengembangan mencakup pemahaman konsep, pembelajaran Greenfoot, implementasi dasar, serta iterasi dan penyesuaian untuk meningkatkan permainan.

Rangkaian Laporan: Laporan ini mencakup tahap-tahap yang merinci deskripsi code, modifikasi, pengujian, dan evaluasi dari implementasi Tetris dalam Greenfoot.

Keseluruhan, struktur kode terdiri dari kelas-kelas yang mewakili elemen-elemen permainan dengan metode-metode yang mengatur perilaku, interaksi, dan tampilan. Setiap kelas memiliki peran spesifik dalam menyusun keseluruhan pengalaman bermain Tetris.