**LAPORAN TUGAS AKHIR PEMROGRAMAN BERORIENTASI OBJEK**

**“HUNGER FISH”**

Diajukan untuk memenuhi tugas mata kuliah Pemrograman Berorientasi Objek yang diampu

Oleh dosen: *Sayekti Harits Suryawan, S.Kom,M.Kom*

****

**Disusun Oleh:**

*Muhamad Reza Vidiansyah (2211102441093)*

*Citro Yoga Pangestu (2211102441248)*

*Aisyah Dinah Zain (2211102441139)*

*M. Fajar (2211102441198)*

**UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR**

**SAMARINDA**

**2023**

**LATAR BELAKANG**

Salah satu konsep yang populer dalam pengembangan game saat ini adalah game berbasis objek. Game berbasis objek mengutamakan penggunaan objek-objek atau elemen-elemen terpisah yang dapat berinteraksi satu sama lain untuk menciptakan pengalaman bermain yang lebih dinamis dan kompleks. Pemodelan objek dalam pengembangan game ini memberikan fleksibilitas, memungkinkan para mahasiswa untuk membuat lingkungan yang realistis, karakter yang kompleks, dan skenario permainan yang lebih kreatif.

Laporan ini akan menggambarkan proses pembuatan game berbasis objek yang mencakup langkah-langkah dari perencanaan , penggunaan element serta implementasinya. Fokus utama dari proyek ini adalah menggali potensi kreatifitas dalam perancangan objek, pemrograman interaksi objek, dan optimasi performa game. Dengan menggunakan metode pengembangan yang terstruktur, diharapkan hasil akhirnya akan menjadi sebuah game yang dapat menjadi gambaran bagi yang lainya.

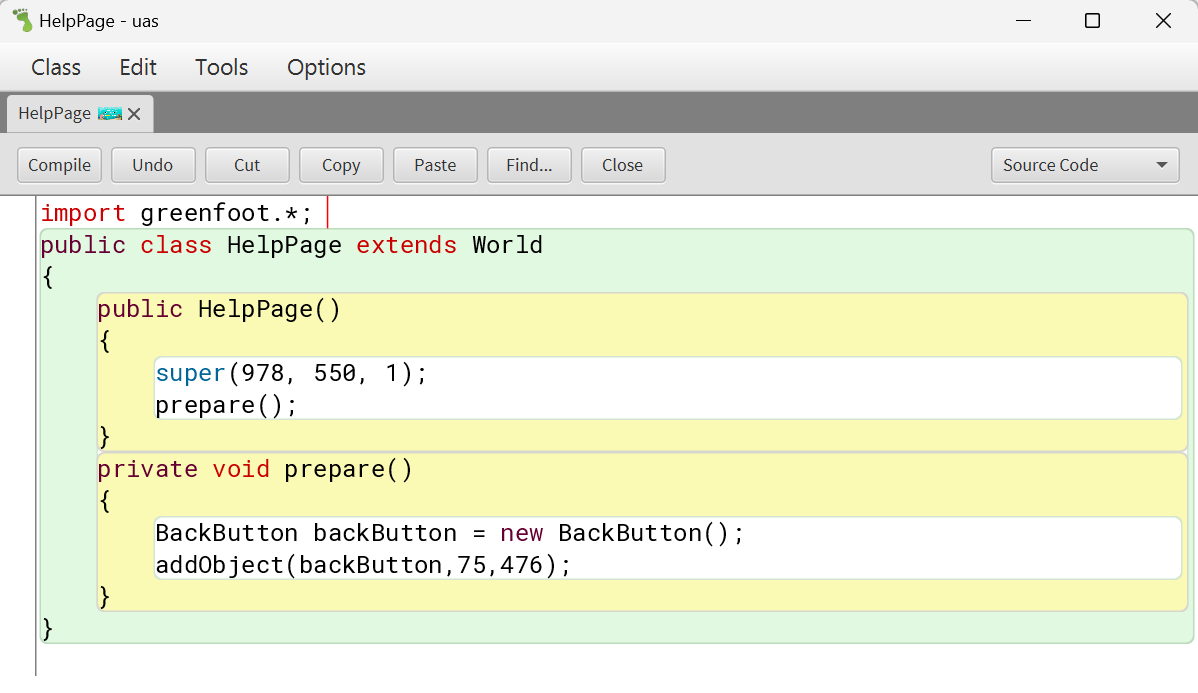
Pembuatan game berbasis objek ini bukan hanya sekedar kegabutan atau apapun, namun juga memperkenalkan kepada para mahasiswa tentang konsep pemrograman berorientasi objek dan logika permainan yang kompleks. Oleh karena itu, laporan ini tidak hanya berfokus pada aspek teknis pengembangan, tetapi juga pada aspek kreatifitas dan pendidikan yang terkandung dalam proses ini.

Dengan dukungan teknologi terkini dan paradigma pengembangan game berbasis objek, diharapkan laporan ini dapat memberikan wawasan yang mendalam mengenai proses dan tantangan yang dihadapi dalam menciptakan pengalaman dalam pengembangan selanjutnya melalui penerapan prinsip-prinsip pengembangan game berbasis objek.

**PEMBAHASAN**

**(WORLD)**

**Help Page**

****

Penjelasan:

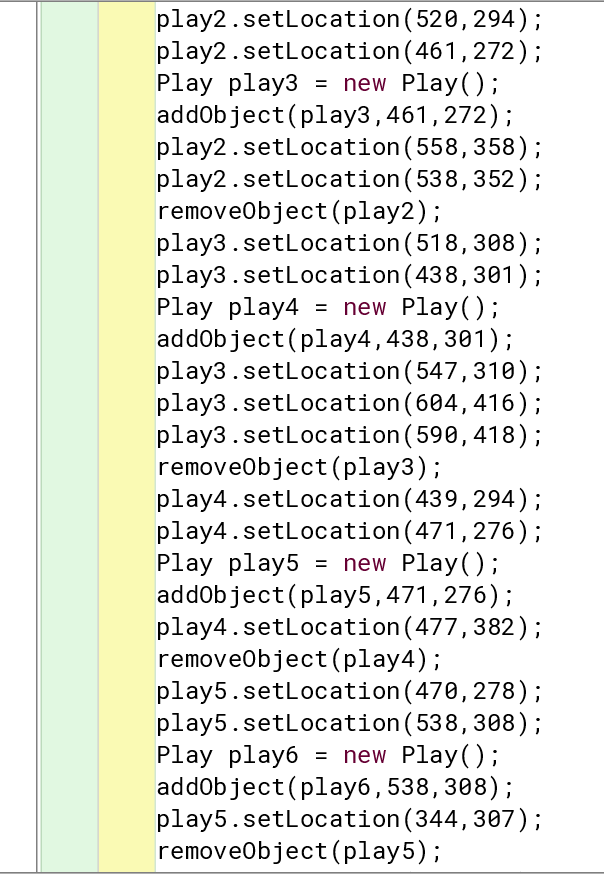
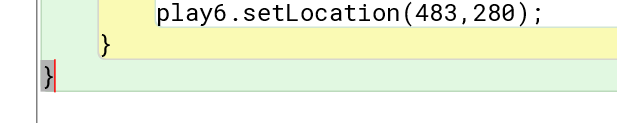
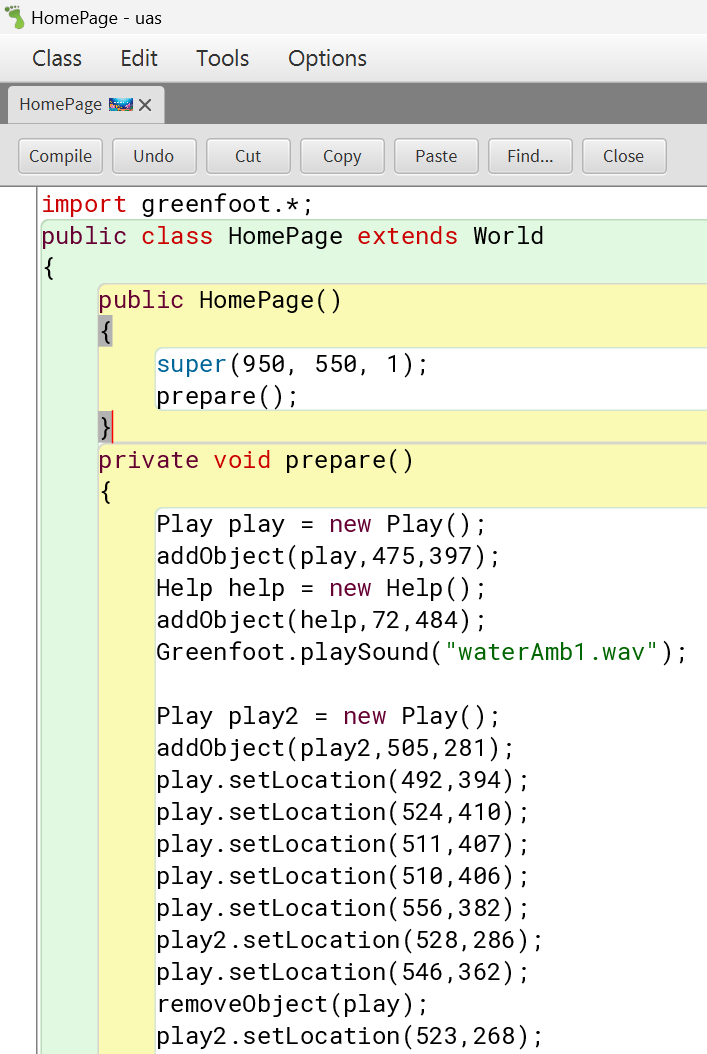
* Konstruktor HelpPage():

Konstruktor inisialisasi objek dunia dengan ukuran 978x550 piksel dan lapisan 1, lalu memanggil metode prepare().

* Metode prepare():

Menyiapkan elemen-elemen dalam dunia, seperti membuat dan menempatkan objek BackButton pada koordinat (75, 476).

* Objek BackButton:
* Membuat objek BackButton, tombol kembal implementasinya tidak terdefinisi dalam potongan kode.
* Menambahkan objek BackButton ke dunia pada koordinat (75, 476), sehingga akan muncul pada posisi tersebut saat program dijalankan.

**Home Page**

Penjelasan:

* Deklarasi Kelas HomePage:

Kelas HomePage adalah turunan dari kelas World dalam Greenfoot, digunakan untuk membuat dunia permainan.

* Konstruktor HomePage:

Konstruktor inisialisasi objek dunia dengan ukuran 950x550 piksel dan lapisan 1, lalu memanggil metode prepare().

* Metode prepare():
* Membuat objek Play dan Help.
* Menambahkan objek Play dan Help ke dalam dunia pada posisi tertentu.
* Memainkan suara dengan file "waterAmb1.wav".
* Melakukan serangkaian perubahan lokasi dan manipulasi objek Play dengan menggunakan metode setLocation() dan removeObject().
* Objek Play dan Help:

Membuat objek Play dan Help, kemungkinan merupakan elemen-elemen interaktif pada halaman utama permainan.

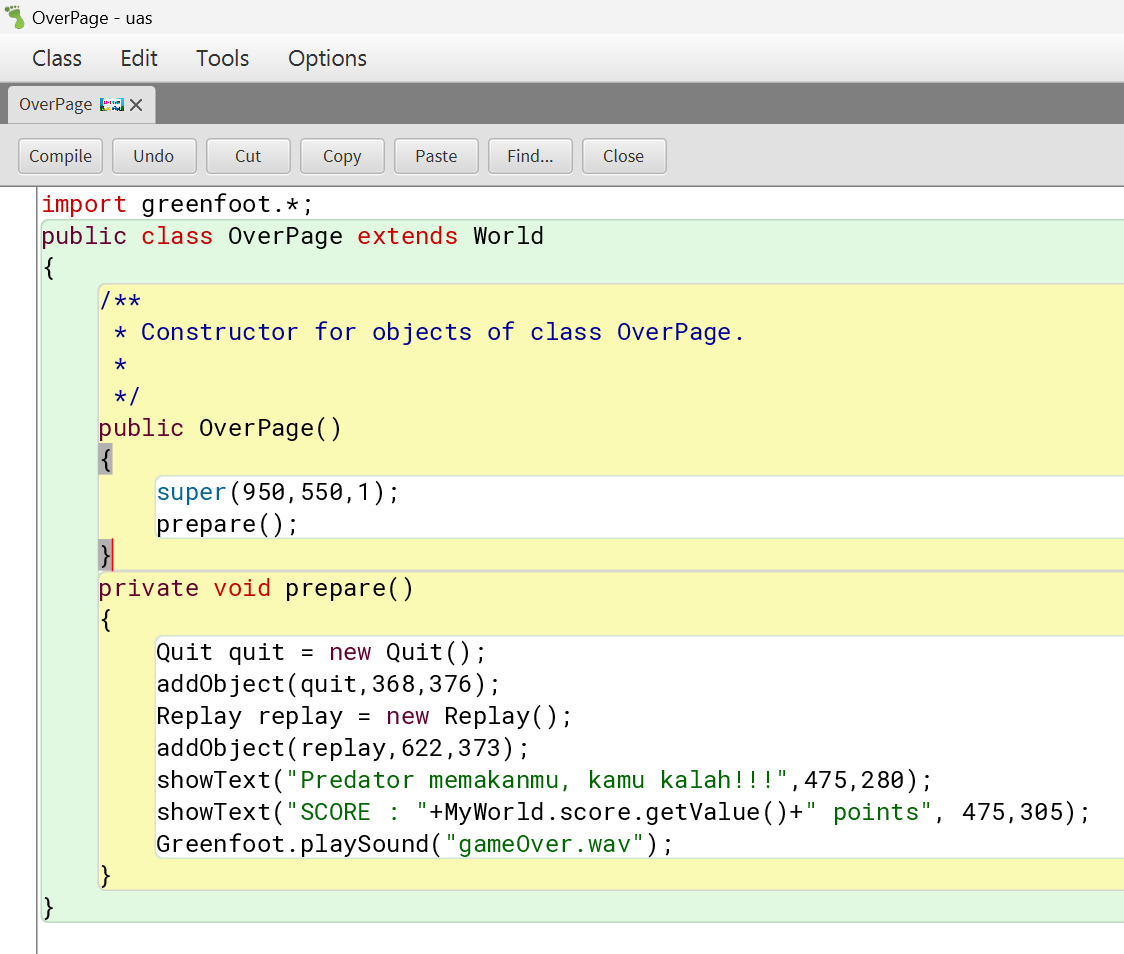
* Menambahkan Objek ke Dunia:

Menambahkan objek Play dan Help ke dalam dunia pada posisi yang telah ditentukan.

* Manipulasi Lokasi Objek Play:

Melakukan perubahan lokasi dan manipulasi objek Play dengan menggunakan berbagai metode, seperti setLocation() dan removeObject().

**Over Page**

****

Penjelasan

* Deklarasi Kelas OverPage:

Kelas OverPage merupakan turunan dari kelas World dalam Greenfoot, digunakan untuk membuat halaman permainan setelah permainan berakhir.

* Konstruktor OverPage:

Konstruktor inisialisasi objek dunia dengan ukuran 950x550 piksel dan lapisan 1, lalu memanggil metode prepare().

* Metode prepare():

Menyiapkan elemen-elemen dalam dunia setelah permainan berakhir. Membuat objek Quit (tombol keluar), Replay (tombol ulang), menampilkan teks "Predator memakanmu, kamu kalah!!!" dan menampilkan skor permainan. Selain itu, memainkan suara "gameOver.wav".

* Objek Quit:

Membuat objek Quit, kemungkinan tombol keluar, namun implementasinya tidak terdefinisi dalam potongan kode.

* Objek Replay:

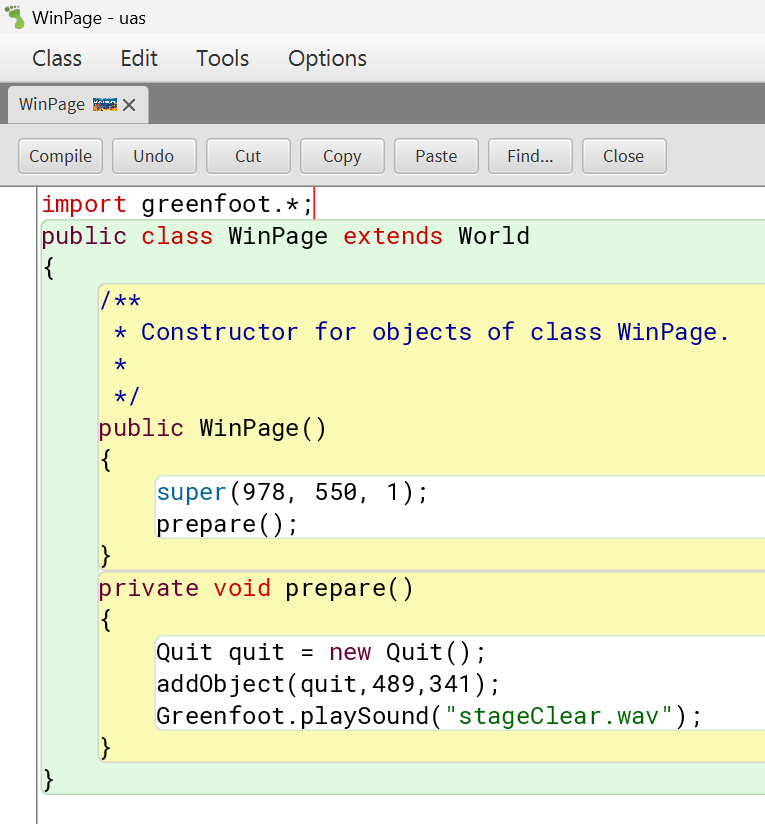
Membuat objek Replay, kemungkinan tombol ulang, namun implementasinya tidak terdefinisi dalam potongan kode.

* Menampilkan Teks:

Menampilkan dua teks, yaitu "Predator memakanmu, kamu kalah!!!" dan "SCORE : [nilai skor] points", dengan skor diambil dari variabel MyWorld.score.

* Membuat Suara:

Memainkan suara "gameOver.wav" saat halaman permainan berakhir.

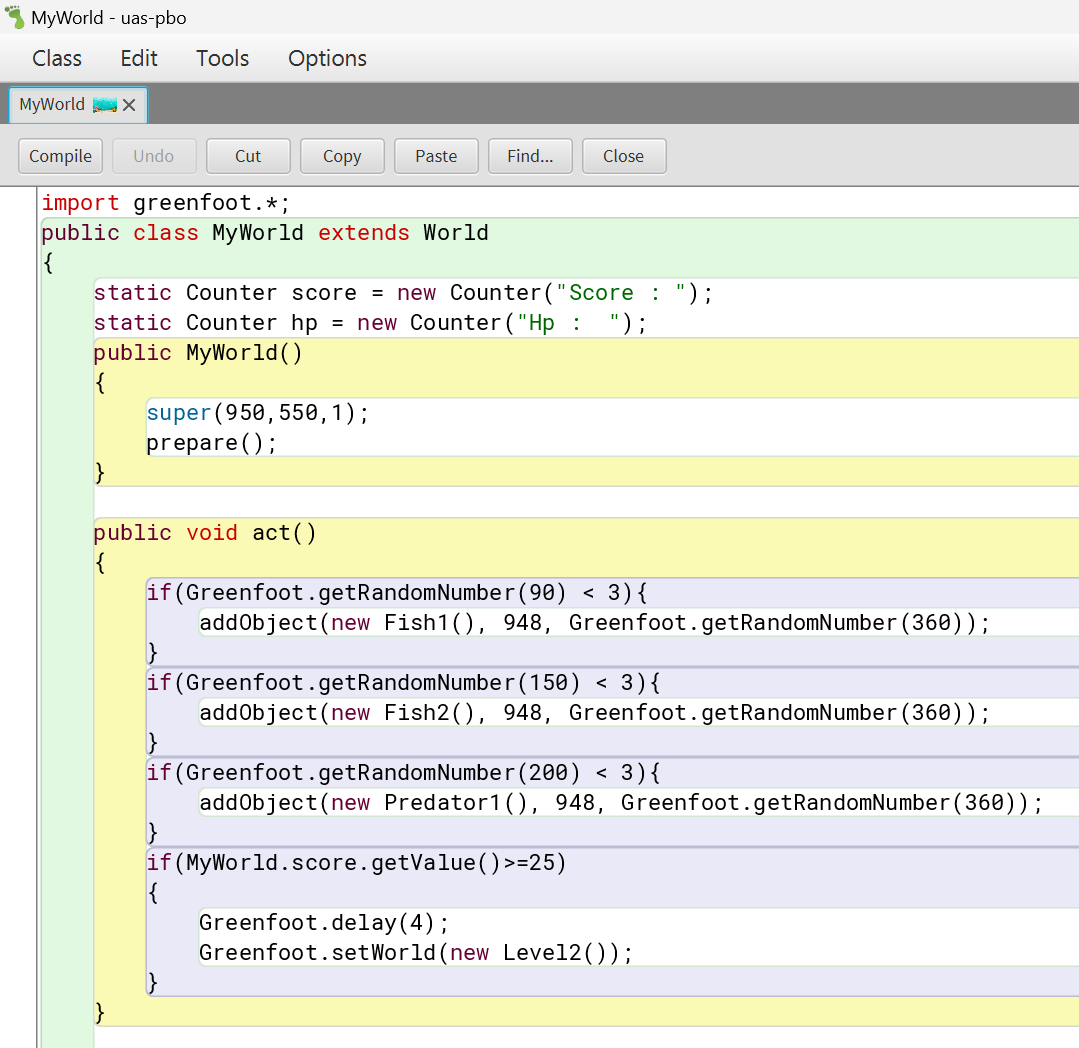
**Win Page**

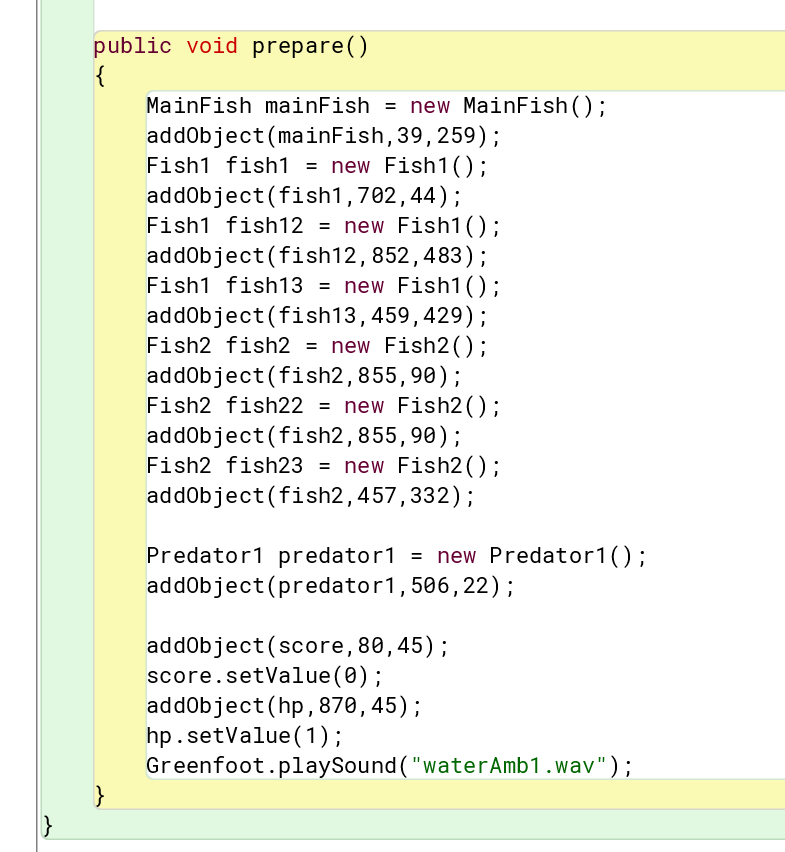
Penjelasan:

* Deklarasi Kelas WinPage:
* Kelas WinPage merupakan turunan dari kelas World di Greenfoot, digunakan untuk membuat dunia permainan.
* Konstruktor WinPage:
* Konstruktor inisialisasi objek dunia dengan ukuran 978x550 piksel dan lapisan 1.
* Memanggil metode prepare() untuk menyiapkan elemen-elemen dalam dunia.
* Metode prepare():
* Membuat objek Quit (kemungkinan tombol keluar) dan menempatkannya pada koordinat (489, 341) dalam dunia.
* Memainkan suara "stageClear.wav" sebagai umpan balik kemenangan.
* Objek Quit:

Membuat objek Quit, yang kemungkinan merupakan tombol keluar atau elemen terkait.

* Menambahkan Objek ke Dunia:
* Menambahkan objek Quit ke dalam dunia pada koordinat (489, 341).

**My World (**level 1)



Penjelasan:

* Counter:

Deklarasi dua objek Counter untuk skor (score) dan poin kesehatan (hp).

* Konstruktor MyWorld:

- Inisialisasi dunia dengan ukuran 950x550 piksel dan lapisan 1.

- Memanggil metode prepare().

* Metode act():

- Logika penambahan objek ikan ke dunia secara acak.

- Jika skor mencapai 50, pindah ke Level2 setelah penundaan 4 detik.

* Metode prepare():

- Penempatan objek utama, ikan, dan predator dalam dunia.

- Inisialisasi skor dan poin kesehatan.

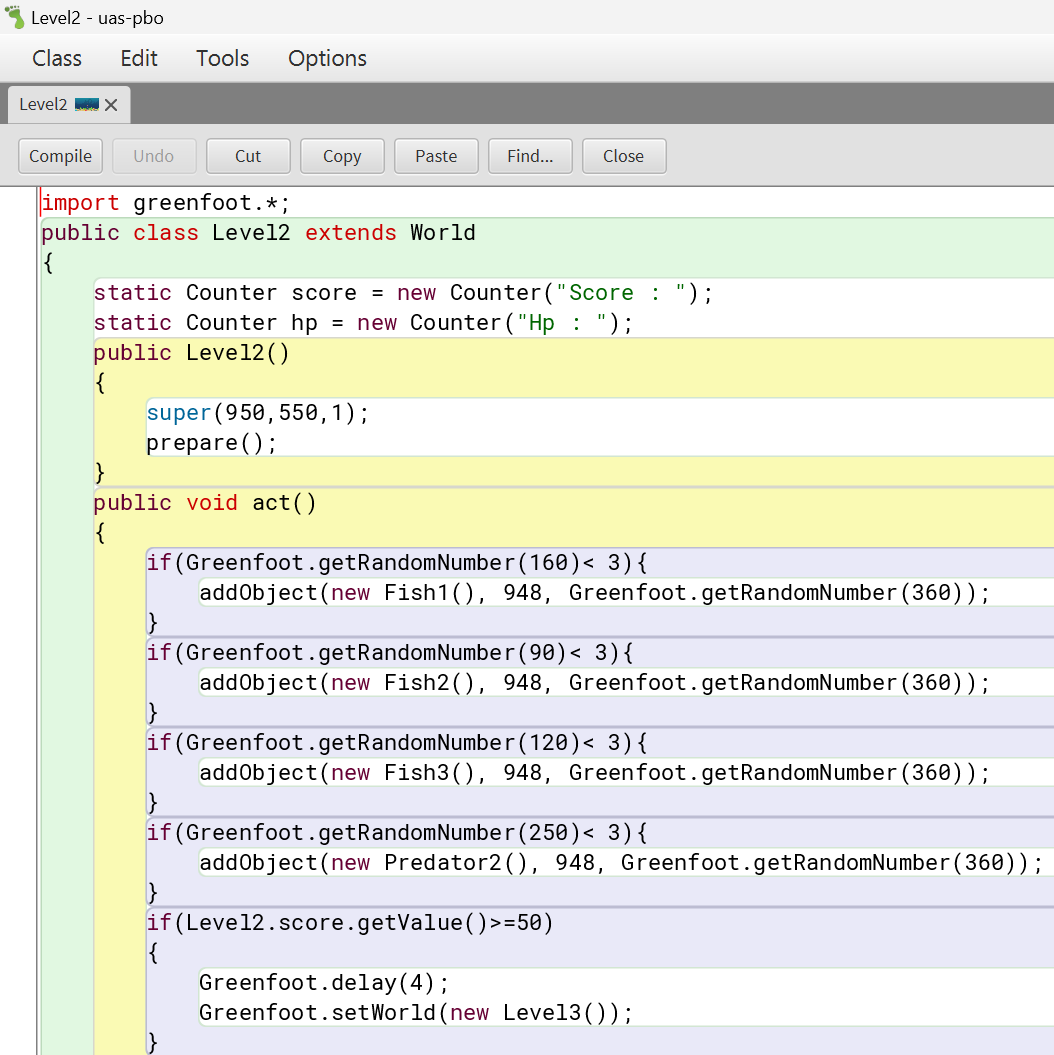
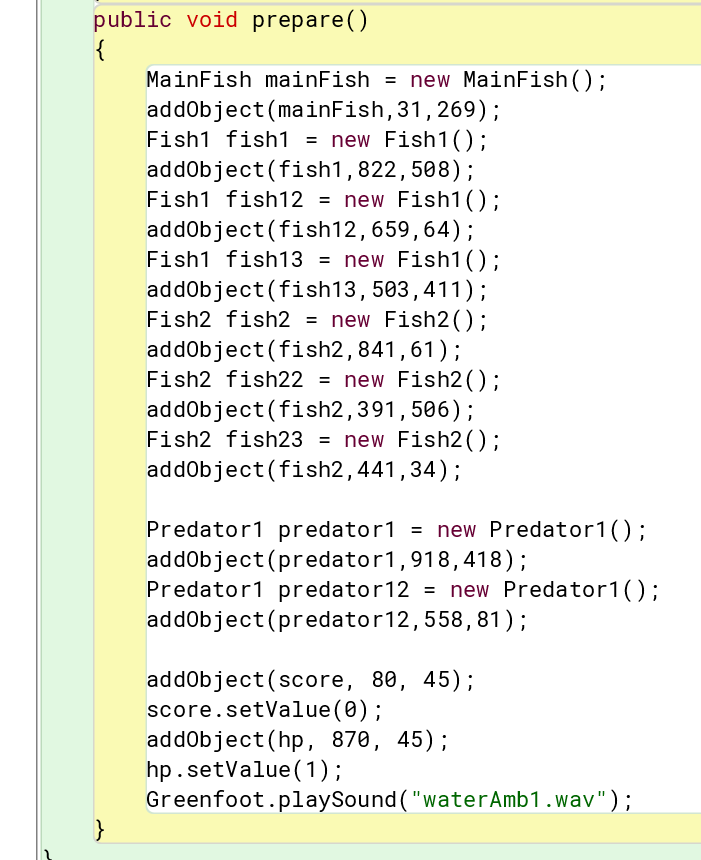
- Memainkan suara latar belakang.

* Penciptaan Objek:

Objek utama (MainFish) dan beberapa objek ikan serta predator.

* Pengecekan Level 2:

Jika skor mencapai 50, program menunggu 4 detik dan pindah ke dunia Level2.

**Level 2**

Penjelasan:

* Deklarasi Kelas Level2:

- Kelas Level2 adalah turunan dari kelas World dalam Greenfoot, mewakili tingkat permainan kedua.

- Membuat dua objek statis Counter untuk skor (score) dan poin kehidupan (hp).

* Konstruktor Level2:

- Konstruktor menginisialisasi dunia dengan ukuran 950x550 piksel dan lapisan 1.

- Memanggil metode prepare() untuk menyiapkan elemen-elemen dalam dunia.

* Metode act():

- Memproses logika permainan setiap frame, termasuk menambahkan objek ikan dan predator secara acak.

- Jika skor mencapai 150, permainan beralih ke Level3 setelah menunda selama 4 frame.

* Metode prepare():

- Menyiapkan elemen-elemen dalam dunia seperti ikan (Fish1, Fish2), predator (Predator1, Predator2), dan objek utama pemain (MainFish).

- Menambahkan objek Counter untuk skor dan poin kehidupan, serta memainkan suara latar belakang.

* Logika Objek dalam act():

- Menggunakan kondisi acak untuk menambahkan objek Fish1, Fish2, Fish3, Predator1, dan Predator2 ke dunia.

- Jika skor mencapai 150, permainan beralih ke Level3 dengan penundaan 4 frame.

* Objek Utama dan Pengaturan Awal:

- Menempatkan objek MainFish pada posisi awal.

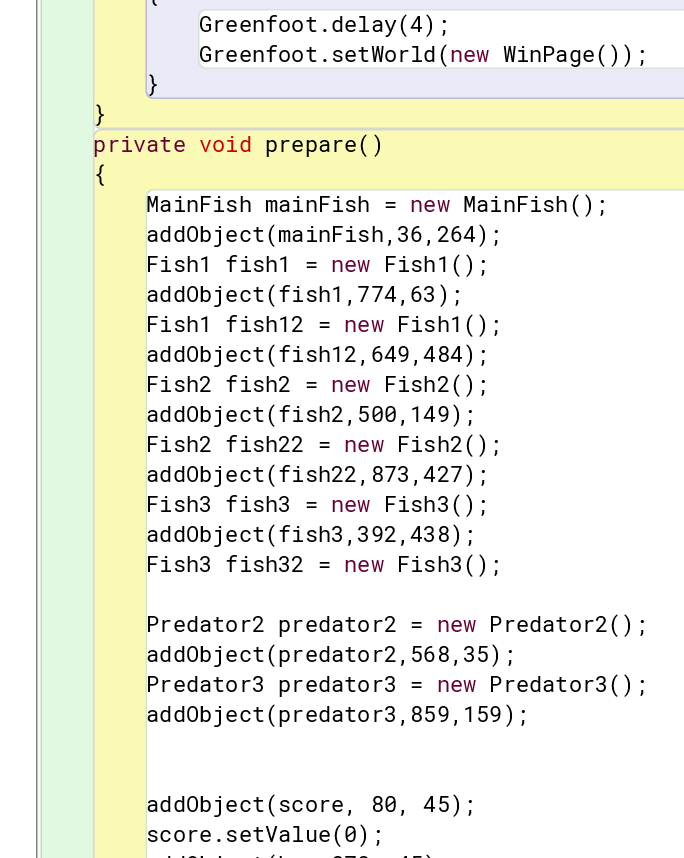
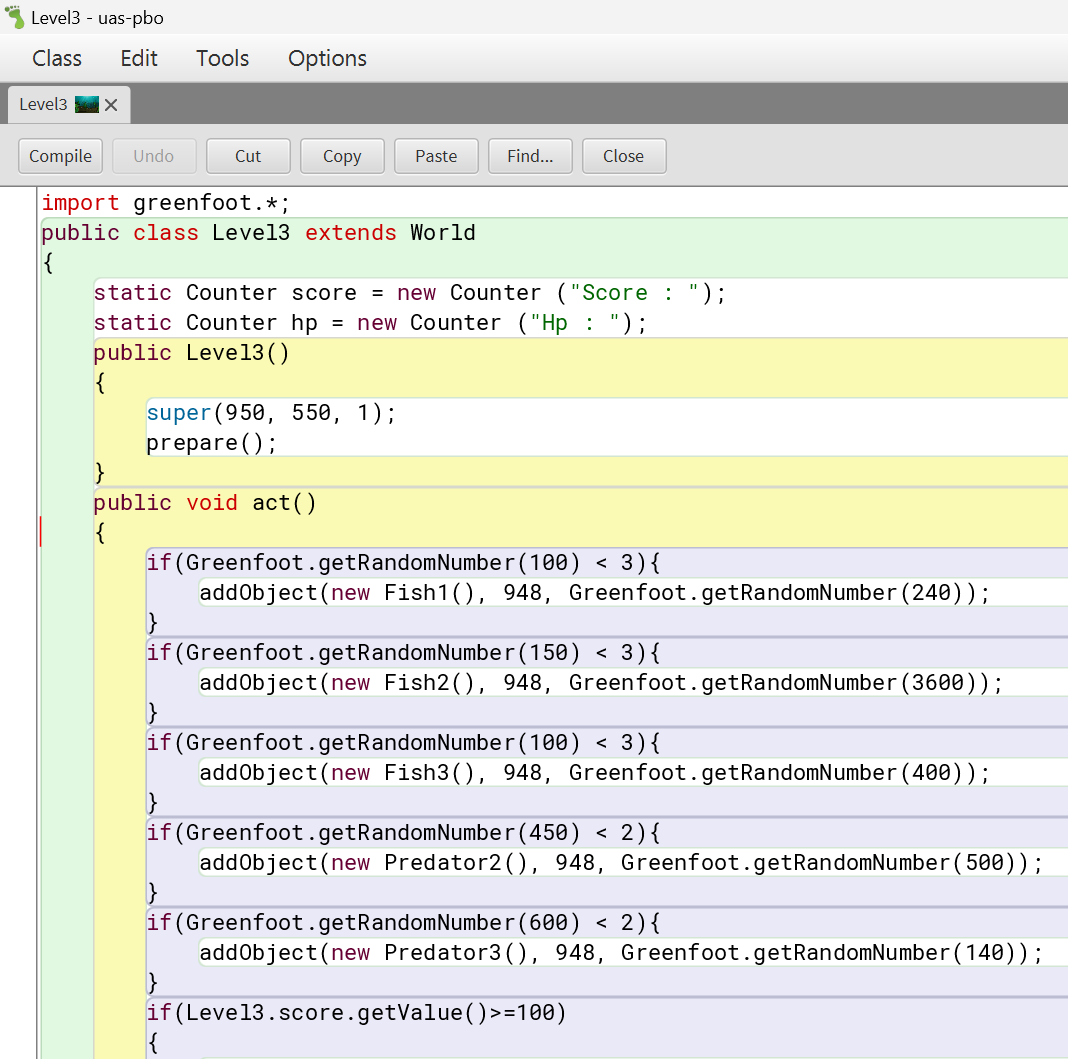
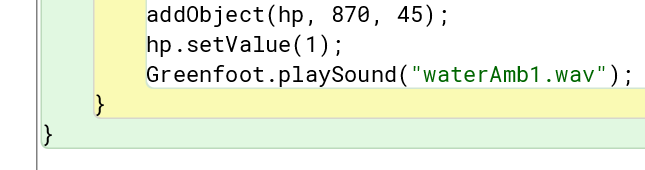
- Menentukan posisi awal beberapa objek Fish1, Fish2, Predator1, dan Predator2.

- Menambahkan objek Counter untuk skor dan poin kehidupan pada posisi tertentu.

- Mengatur nilai awal skor dan poin kehidupan.

* Suara Latar Belakang:

- Memainkan suara latar belakang dengan nama file "waterAmb1.wav" saat memulai permainan.

**Level 3**

Penjelasan:

* Deklarasi Kelas Level3:

Kelas `Level3` merupakan turunan dari kelas `World` dalam Greenfoot dan digunakan untuk mengimplementasikan level ketiga dari permainan.

* Deklarasi Counter Score dan Hp:

Menggunakan dua objek Counter statis, yaitu `score` dan `hp`, untuk melacak nilai skor dan nyawa dalam permainan.

* Konstruktor Level3:

Konstruktor inisialisasi objek dunia dengan ukuran 950x550 piksel dan lapisan 1, lalu memanggil metode `prepare()` untuk menyiapkan elemen-elemen dalam dunia.

* Metode act():

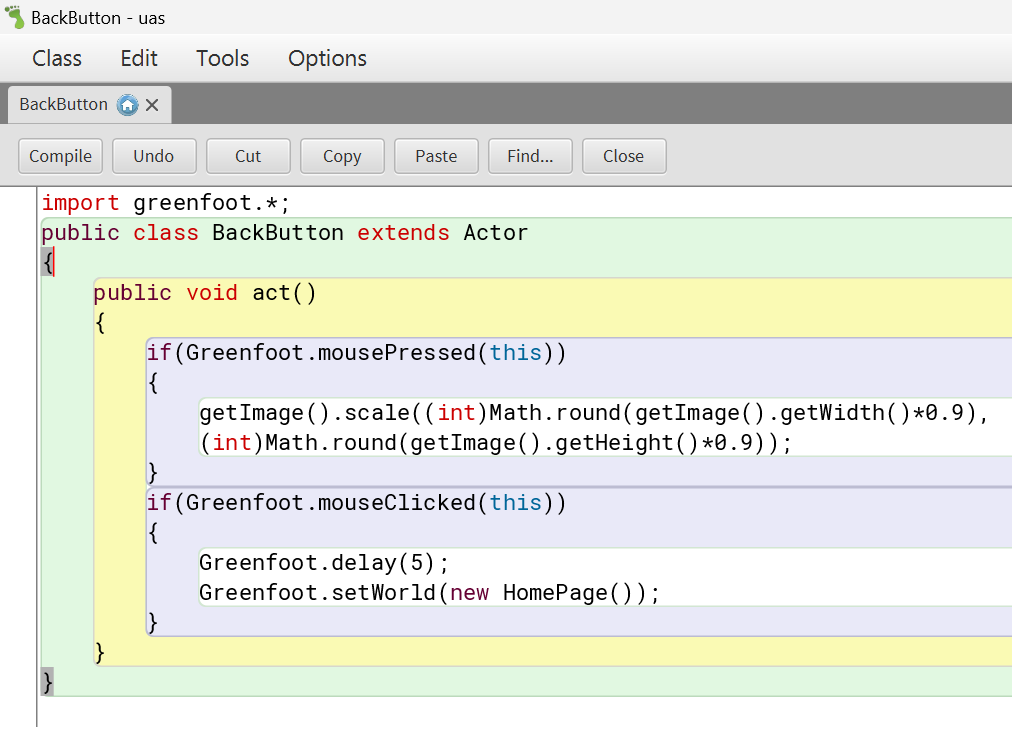
Metode ini berisi logika pergerakan dan penambahan objek seperti `Fish1`, `Fish2`, `Fish3`, `Predator1`, `Predator2`, dan `Predator3` berdasarkan kondisi tertentu. Jika nilai skor mencapai 250, permainan akan pindah ke `WinPage`.

* Metode prepare():

Metode ini menyiapkan elemen-elemen awal dalam dunia seperti objek `MainFish`, `Fish1`, `Fish2`, `Fish3`, `Predator1`, `Predator2`, dan `Predator3`. Juga menambahkan objek Counter `score` dan `hp`, serta memainkan suara latar belakang.

**(ACTOR)**

**BackButton**

****

Penjelasan:

* Deklarasi Kelas BackButton:

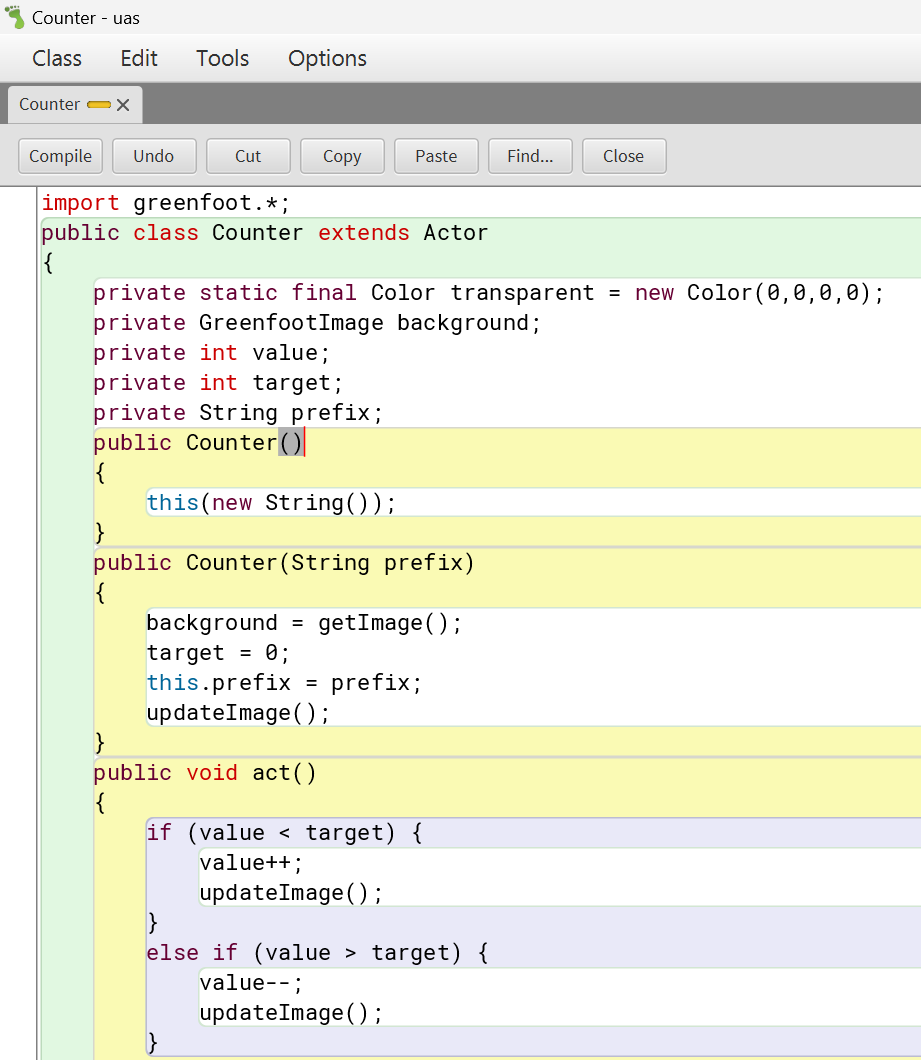
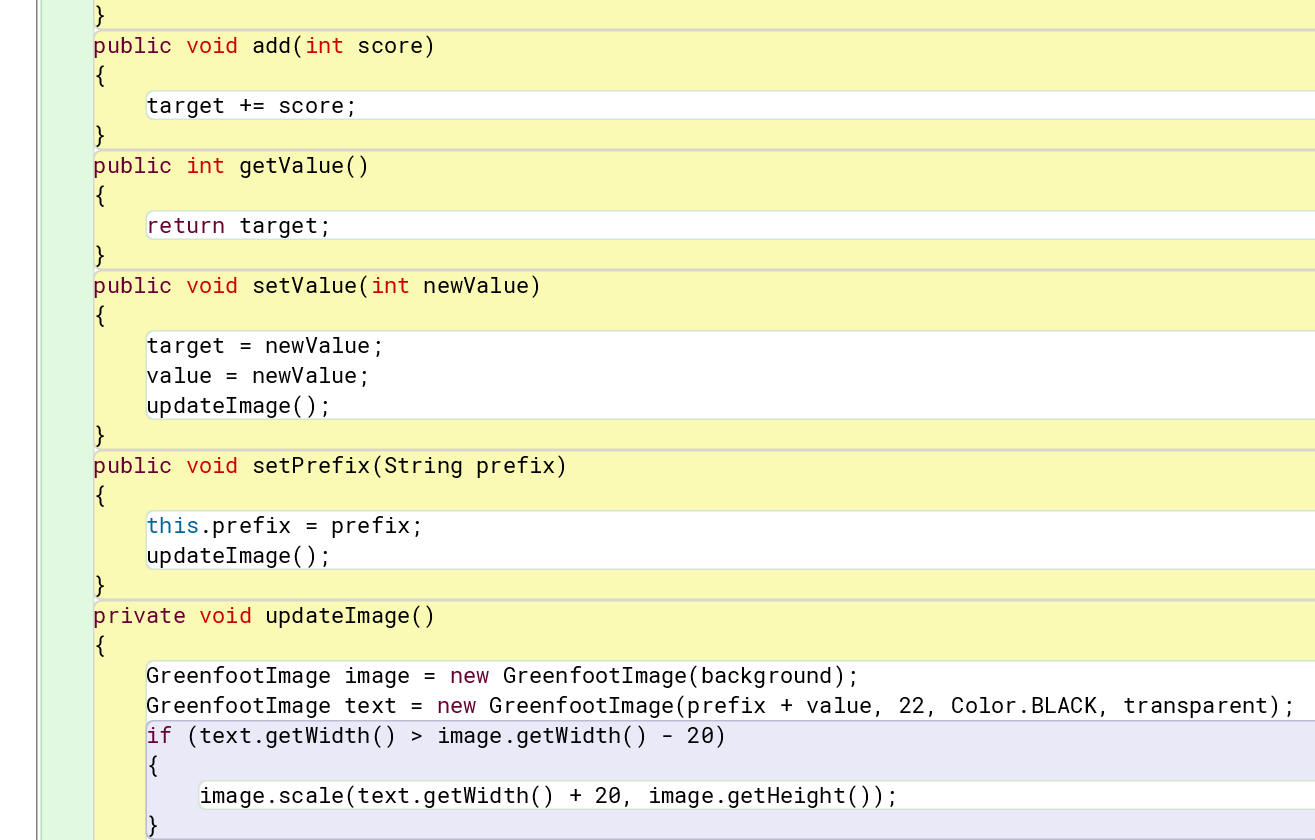
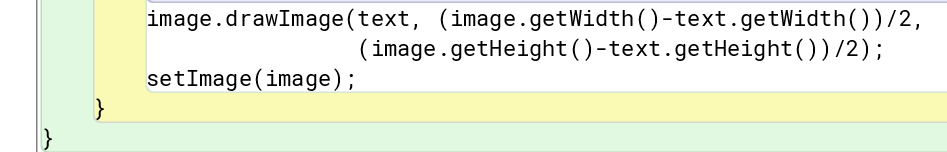
Kelas BackButton adalah turunan dari kelas Actor dalam Greenfoot, menangani perilaku tombol kembali.

* Respons saat Ditekan:

Mengurangi skala tombol sebesar 10% saat ditekan.

* Pergantian Dunia saat Diklik:

Jika tombol diklik, terjadi penundaan sebentar dan dunia digantikan dengan HomePage

**Counter**

Penjelasan:

* Deklarasi Kelas Counter:

Kelas `Counter` merupakan turunan dari kelas `Actor` dalam Greenfoot, digunakan untuk membuat objek hitungan (counter) yang dapat ditampilkan di dunia permainan.

* Variabel dan Objek:

- private static final Color transparent: Variabel untuk warna transparan.

- private GreenfootImage background: Objek untuk menyimpan gambar latar belakang.

- private int value: Variabel menyimpan nilai hitungan.

- private int target: Variabel menyimpan nilai target hitungan.

- private String prefix: Variabel menyimpan awalan teks.

* Konstruktor Counter:

- Terdapat dua konstruktor, satu tanpa parameter dan satu dengan parameter `prefix`.

- Menginisialisasi variabel, menetapkan gambar latar belakang dari objek, dan memanggil metode updateImage()`.

* Metode act():

- Diaktifkan setiap frame.

- Menambah atau mengurangkan nilai hitungan dan memperbarui gambar sesuai kondisi.

* Metode add(int score):

- Menambahkan nilai `score` ke nilai target hitungan.

* Metode getValue():

- Mengembalikan nilai target hitungan.

* Metode setValue(int newValue):

- Menetapkan nilai target dan nilai aktual hitungan, lalu memperbarui gambar.

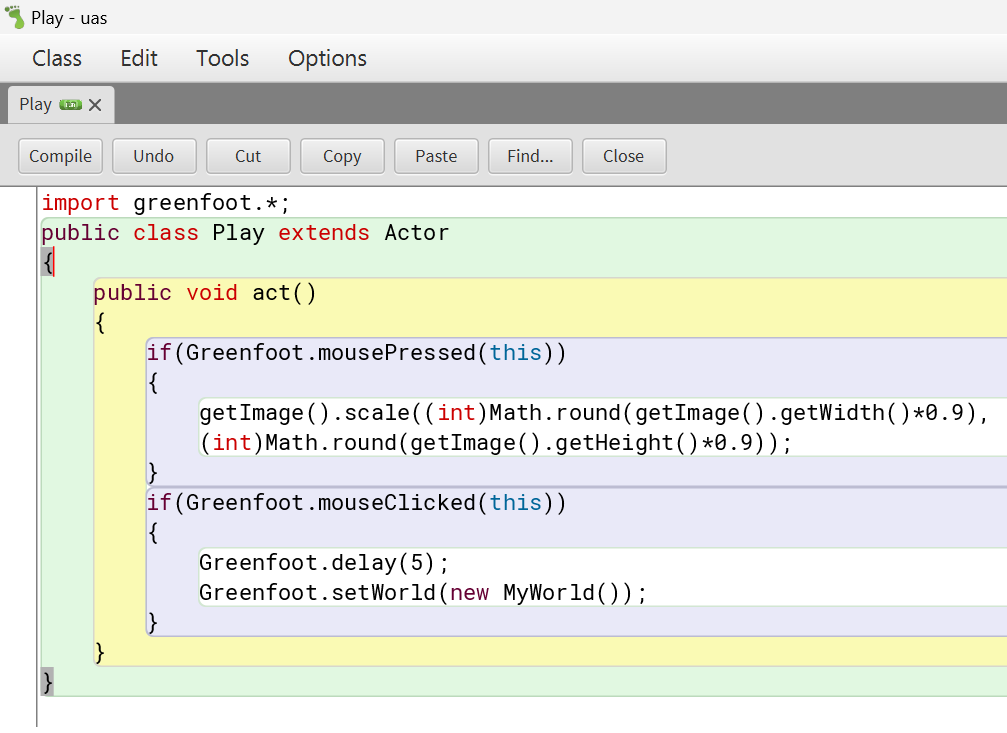
* Metode setPrefix(String prefix):

- Menetapkan awalan teks dan memperbarui gambar.

* Metode updateImage():

- Memperbarui gambar hitungan dengan menyesuaikannya sesuai nilai dan teks.

- Mengatur ulang ukuran gambar jika teks melebihi lebar gambar latar belakang.

**Play 1**

Penjelasan:

* Deklarasi Kelas Play:

Kelas `Play` adalah turunan dari `Actor` dalam Greenfoot, berfungsi sebagai objek yang dapat berinteraksi di dunia permainan.

* Metode act():

Metode `act()` menentukan aksi objek `Play` pada setiap siklus permainan.

* Periksa Pencetan Mouse:
* Mengecek apakah tombol mouse ditekan pada objek `Play`, dan jika ya, mengubah ukuran gambar menjadi 90% dari ukuran sebelumnya.
* Periksa Klik Mouse:

Mengecek apakah objek `Play` di-klik; jika ya, menunda 5 siklus dan mengganti dunia permainan dengan objek `MyWorld` yang baru.

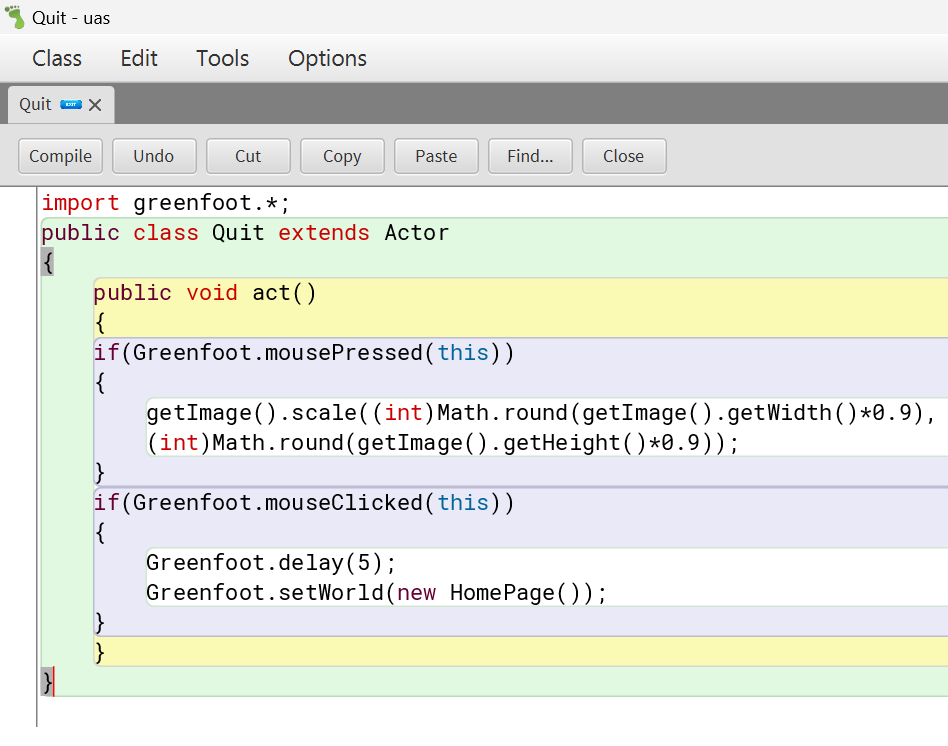
* Mengubah Ukuran Gambar:

Mengubah ukuran gambar objek `Play` menjadi 90% dari ukuran sebelumnya saat tombol mouse ditekan.

* Pergantian Dunia:

Mengganti dunia permainan dengan membuat objek baru dari kelas `MyWorld` saat objek `Play` di-klik.

**Quit**

****

Penjelasan:

* Kelas Quit:

Mewakili kelas `Quit`, merupakan turunan dari kelas `Actor` dalam Greenfoot. Kelas ini berperan sebagai objek yang dapat berinteraksi dalam dunia permainan.

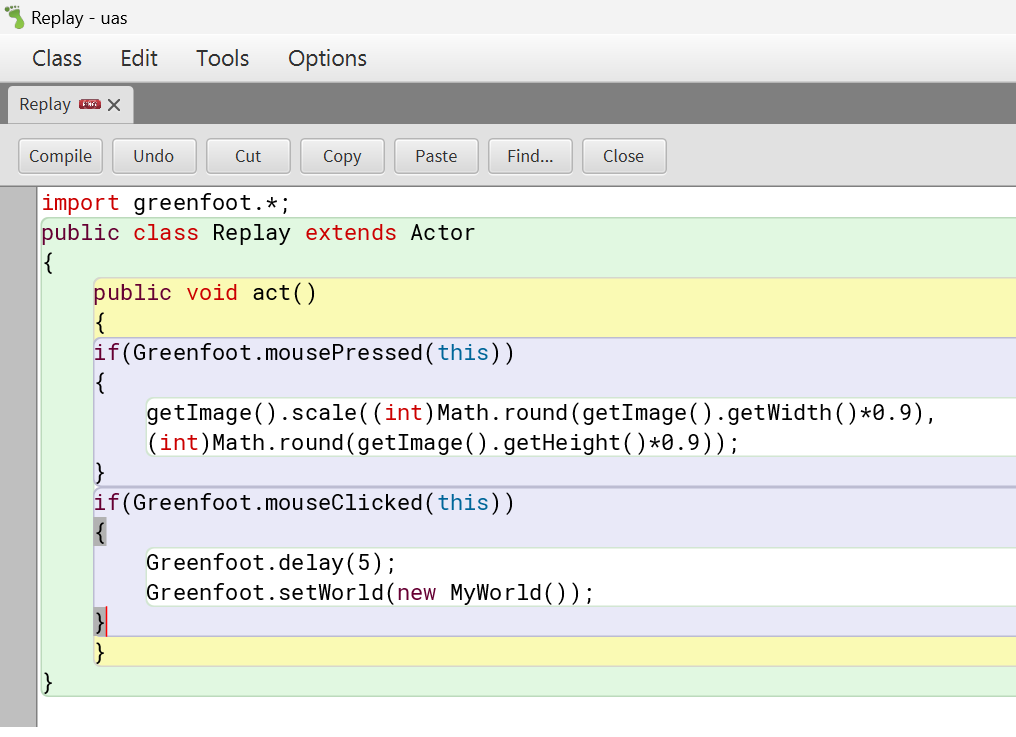
* Metode act():
* Metode `act()` adalah metode khusus dalam Greenfoot yang dieksekusi secara terus-menerus. Dalam konteks kelas ini, metode ini berisi logika interaksi objek.
* Pengecekan Tombol Ditekan:

Jika objek `Quit` ditekan (mousePressed), gambar objek akan diperkecil sebesar 90% dari ukuran aslinya.

* Pengecekan Tombol Diklik:

Jika objek `Quit` diklik (mouseClicked), maka akan terjadi delay selama 5 frame (Greenfoot.delay(5)), kemudian dunia akan diubah menjadi `HomePage` menggunakan `Greenfoot.setWorld(new HomePage())`. Ini mengindikasikan bahwa objek `Quit` berfungsi sebagai elemen untuk kembali ke halaman utama atau keluar dari permainan.

**Replay**

****

Penjelasan:

* Deklarasi Kelas Replay:

Mendefinisikan kelas `Replay` yang merupakan turunan dari kelas `Actor` dalam lingkungan pengembangan permainan Greenfoot. Kelas `Actor` digunakan untuk objek dalam dunia permainan.

* Metode act():

- Metode `act()` merupakan metode yang dieksekusi secara berulang oleh Greenfoot untuk melakukan aksi-aksi tertentu pada objek `Replay`.

- Pertama, melakukan pengecekan apakah tombol mouse ditekan pada objek `Replay`.

- Jika ya, gambar objek diperkecil sebesar 90% dari ukuran sebelumnya.

- Selanjutnya, melakukan pengecekan apakah tombol mouse diklik pada objek `Replay`.

- Jika ya, terjadi penundaan selama 5 frame menggunakan `Greenfoot.delay(5)`.

- Setelah penundaan, dunia permainan diubah menjadi dunia baru (`MyWorld`) melalui `Greenfoot.setWorld(new MyWorld())`.

* Interaksi dengan Mouse:

- Menggunakan `Greenfoot.mousePressed(this)` untuk mendeteksi ketika mouse ditekan pada objek `Replay`.

- Menggunakan `Greenfoot.mouseClicked(this)` untuk mendeteksi ketika mouse diklik pada objek `Replay`.

* Pengubahan Ukuran Gambar:

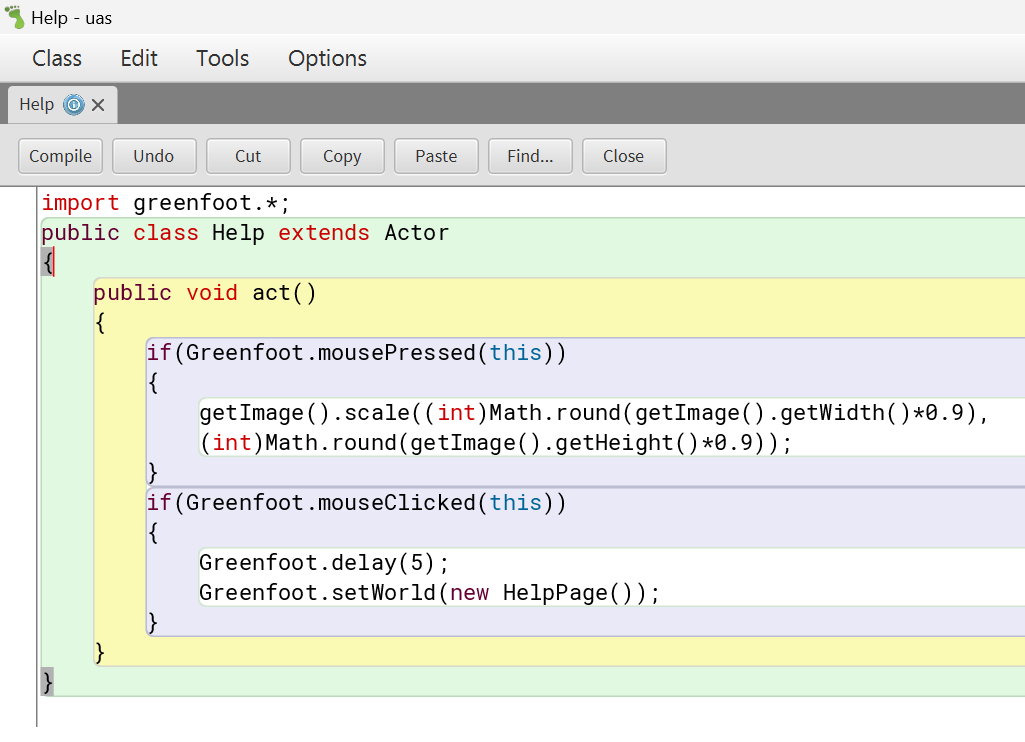
- Menggunakan `getImage().scale()` untuk mengubah ukuran gambar objek `Replay`.

- Perubahan ukuran dilakukan sebesar 90% dari ukuran sebelumnya.

* Restart Permainan:

- Jika objek `Replay` diklik, terjadi penundaan selama 5 frame sebelum dunia permainan diubah menjadi dunia baru (`MyWorld`) dengan menggunakan `Greenfoot.setWorld(new MyWorld())`.

**Help**

****

Penjelasan:

* Deklarasi Kelas Help:

Mendeklarasikan kelas `Help` sebagai turunan dari kelas `Actor` dalam Greenfoot, mungkin sebagai representasi objek atau karakter dalam permainan.

* Metode act():

Menentukan aksi atau perilaku objek `Help` saat berinteraksi.

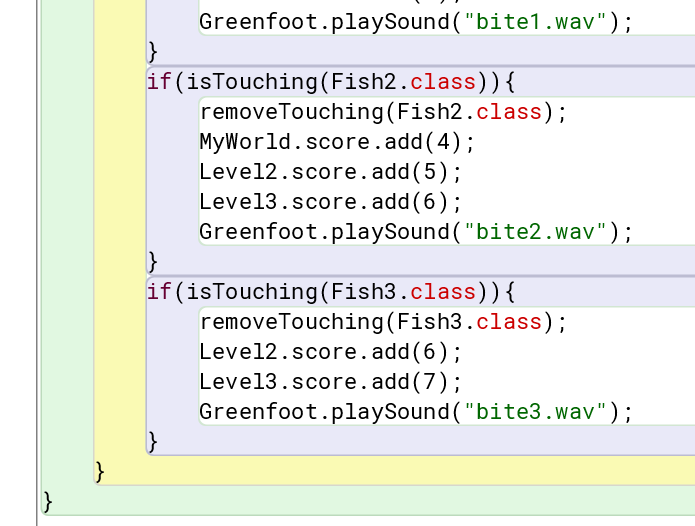
* Mouse Ditekan:

Jika mouse ditekan pada objek `Help`, ukuran gambar dikurangi menjadi 90%.

* Mouse Diklik:

Jika objek `Help` diklik, terjadi penundaan 5 frame, dan dunia permainan diubah menjadi `HelpPage`.

**Main Fish**

****Penjelasan:

* Kelas MainFish:

Merupakan kelas yang merupakan turunan dari kelas `Actor` dalam Greenfoot, digunakan untuk mengontrol perilaku ikan utama dalam permainan.

* Metode act():

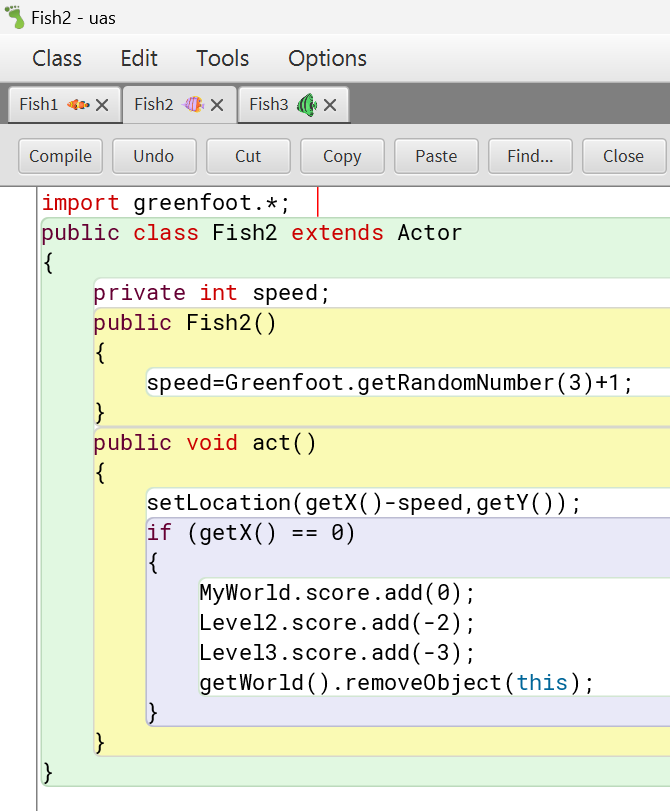
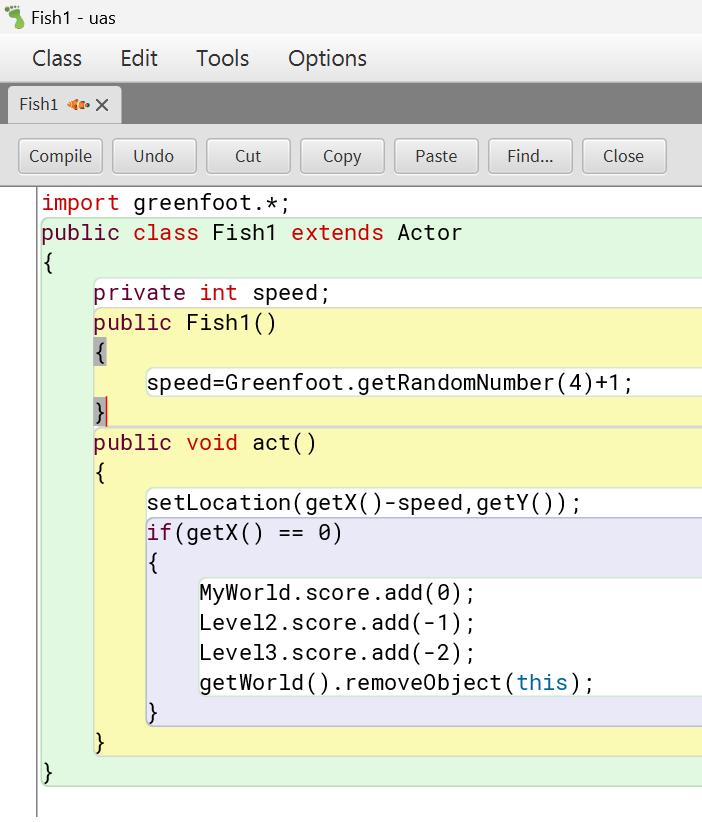
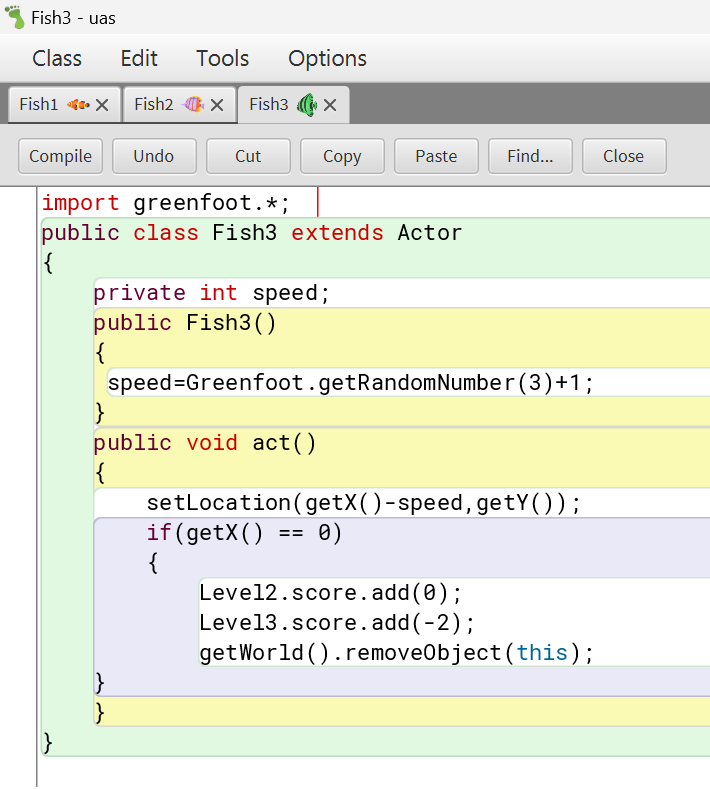
Merupakan metode utama yang dipanggil secara berulang selama jalannya permainan. Metode ini memanggil dua metode lain, yaitu `checkKeyPress()` untuk memeriksa input keyboard dan `checkCollision()` untuk memeriksa tabrakan dengan ikan lain.

* Metode checkKeyPress():

Memeriksa tombol yang ditekan oleh pemain (up, down, left, right) dan memindahkan lokasi ikan utama sesuai dengan input tersebut.

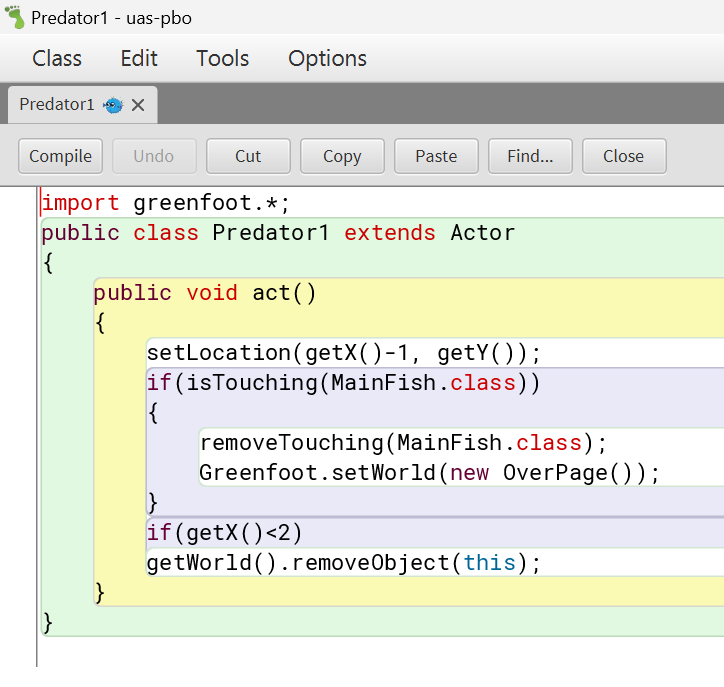
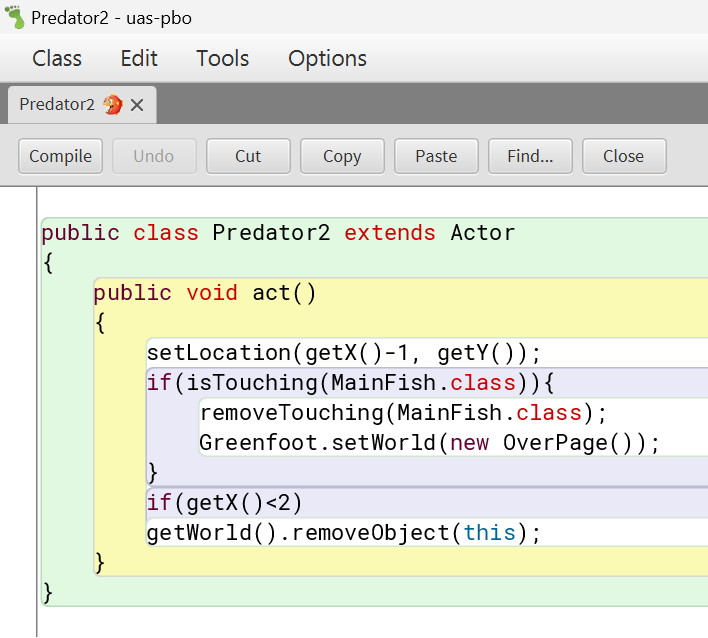
* Metode checkCollision():

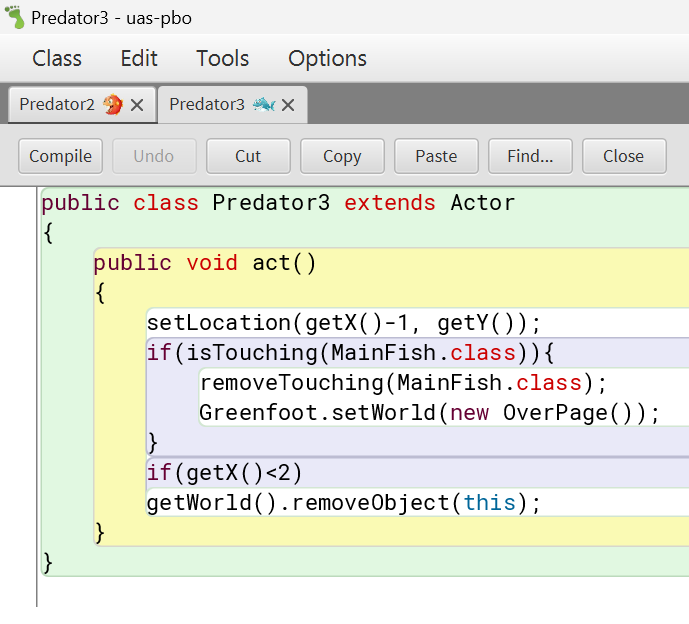
Memeriksa tabrakan ikan utama dengan ikan-ikan lain dalam permainan. Jika terjadi tabrakan dengan ikan jenis tertentu (Fish1, Fish2, Fish3), maka ikan tersebut dihapus, skor pemain dan skor level (Level2 dan Level3) ditingkatkan, dan suara efek (bite1.wav, bite2.wav, bite3.wav) diputar sesuai dengan jenis ikan yang ditangkap.

**Fish 1, 2 dan 3**

Penjelasan:

* Fish1 Class:
  + Kelas Fish1 adalah turunan dari kelas Actor dalam Greenfoot, merepresentasikan ikan dengan kecepatan acak.
  + Konstruktor menetapkan kecepatan dengan nilai acak antara 1 hingga 4.
  + Metode act() menggerakkan ikan ke kiri sejauh kecepatan yang telah ditetapkan. Jika mencapai batas layar (getX() == 0), nilai skor pada berbagai level diubah, dan objek ikan dihapus dari dunia.
* Fish2 Class:
  + Kelas Fish2 juga merupakan turunan dari kelas Actor dan merepresentasikan jenis ikan lainnya dengan kecepatan acak.
  + Konstruktor menetapkan kecepatan dengan nilai acak antara 1 hingga 3.
  + Metode act() menggerakkan ikan ke kiri dengan kecepatan yang telah ditetapkan. Jika mencapai batas layar, nilai skor diubah, dan objek ikan dihapus dari dunia.
* Fish3 Class:
  + Kelas Fish3 adalah turunan dari kelas Actor, mewakili jenis ikan ketiga dengan kecepatan acak.
  + Konstruktor menetapkan kecepatan dengan nilai acak antara 1 hingga 3.
  + Metode act() menggerakkan ikan ke kiri dengan kecepatan yang telah ditetapkan. Jika mencapai batas layar, nilai skor diubah pada beberapa level, dan objek ikan dihapus dari dunia.
* Perbedaan:
  + Tipe Ikan dan Kecepatan:
    - Fish1 memiliki kecepatan acak antara 1 hingga 4.
    - Fish2 memiliki kecepatan acak antara 1 hingga 3.
    - Fish3 memiliki kecepatan acak antara 1 hingga 3.
  + Perubahan Skor:
    - Fish1 mengubah nilai skor pada level 1, level 2, dan level 3.
    - Fish2 mengubah nilai skor pada level 1, level 2, dan level 3, dengan nilai yang berbeda dari Fish1.
    - Fish3 mengubah nilai skor pada level 2 dan level 3, dengan nilai yang berbeda dari Fish1 dan Fish2.
  + Level Pemrosesan Skor:
    - Fish1 mengakses nilai skor pada level 1, level 2, dan level 3.
    - Fish2 mengakses nilai skor pada level 1, level 2, dan level 3.
    - Fish3 mengakses nilai skor pada level 2 dan level 3.

**Predator 1, 2 dan 3**

****

Penjelasan:

* Predator1 Class:
  + Kelas ini mengextends kelas Actor, menunjukkan bahwa objek ini dapat ditampilkan dalam dunia Greenfoot.
  + Metode act() mengatur pergerakan objek ke kiri (getX()-4) setiap frame.
  + Jika objek menyentuh objek dari kelas MainFish, objek tersebut dihapus dan dunia diganti dengan OverPage.
  + Jika posisi objek di sumbu x kurang dari 5, objek dihapus dari dunia.
* Predator2 Class:
  + Sama dengan Predator1 namun dengan kecepatan pergerakan yang sedikit lebih lambat (getX()-3).
  + Ketika menyentuh objek dari kelas MainFish, objek tersebut dihapus dan dunia diganti dengan OverPage.
  + Jika posisi objek di sumbu x kurang dari 5, objek dihapus dari dunia.
* Predator3 Class:
  + Sama dengan Predator1 dengan perbedaan kecepatan pergerakan yang mirip dengan Predator1, dan jika posisi objek di sumbu x kurang dari 3, objek dihapus dari dunia.
  + Proses menyentuh objek dari kelas MainFish dan penggantian dunia pada kondisi tertentu mirip dengan Predator1 dan Predator2.
* Perbedaan:
  + Perbedaan utama adalah pada kecepatan pergerakan masing-masing predator: Predator1 bergerak paling cepat, diikuti oleh Predator2, dan Predator3 yang paling lambat.
  + Selain itu, terdapat perbedaan nilai pada kondisi pengecekan posisi (getX()<5 atau getX()<3) yang menentukan kapan predator dihapus dari dunia.

**KESIMPULAN**

laporan ini memberikan gambaran mendalam tentang proses pembuatan game berbasis objek yang melibatkan langkah-langkah perencanaan, penggunaan elemen, dan implementasi berbagai kelas dan objek dalam lingkungan Greenfoot. Proyek ini tidak hanya menekankan aspek teknis pengembangan game, tetapi juga menyoroti aspek kreativitas dan pendidikan yang terkandung dalam proses tersebut. Melalui penggunaan konsep pemrograman berorientasi objek, logika permainan yang kompleks, dan penggunaan suara serta efek, proyek ini menciptakan pengalaman bermain yang dinamis dan menarik. Dengan fokus pada potensi kreativitas, laporan ini memberikan wawasan yang mendalam tentang tantangan dan proses dalam mengembangkan game berbasis objek, menjadikannya sumber inspirasi dan panduan bagi pengembang game yang ingin memahami dan mengimplementasikan prinsip-prinsip pengembangan game berbasis objek.