Snake

Dokumentacija za projekt: Snake

Razvoj softvera

Senida Smajlović, TKN, 5425/M

Senida Smajlović, 5425/M
PMF, 2020.

O igri

Snake je poznata videoigra u kojoj igrač upravlja linijom čija dužina raste. Koncept igre potiče od “Arcade Game Blockade” iz 1976. Jednostavnost implementacije Snake dovela je do stotina različitih verzija za razne platforme. Nakon što je jedna varijanta omogućena za Nokia mobilni telefon 1998., interes za ovu igru je porastao. Postoji preko 300 različitih igri poput ove samo za iOS.

Igrač kontroliše tačku, kvadrat, ili bilo koji drugi dati objekat. Kako se pomiče naprijed, iza sebe ostavlja trag, što podsjeća na zmiju (snake) koja se kreće. U nekim varijantama igre zadnji dio zmije je fiksiran, tako da se zmija stalno povećava kako se pomjera. S druge strane, postoje varijante u kojima zmija ima određenu dužinu, koja raste samo kada “pojede” tj. pređe neku zadanu prepreku. Igrač gubi kada udari u zid ili dio svog “tijela”.

O projektu

Projekat “Snake” rađen je u programskom jeziku java.
U ovom projektu omogućeno je da igra jedan igrač. Zmija na početku ima već određenu dužinu, i ima određenu početnu tačku od koje kreće igra. Na početku se postavi, na random poziciji, “dozvoljena prepreka”, i cilj je da zmija pokupi tu prepreku, što će joj povećati dužinu za jedan, a prepreka će biti postavljena na novo random mjesto.
Dakle, cilj igre je da igrač, pomjerajući zmiju pokupi što više “dozvoljenih prepreka” od čega dužina zmije raste. Igrač gubi onda kada zmija udari u zid ili dio svoga tijela. Zid je ustvari predstavljen krajevima polja u kojem je igrica postavljena.

Ovaj projekt sastoji se od tri package-a, a svaki od package-a ima svoje klase:

- GUI
  - Frame
  - Game
- Konzola
  - Console
  - ConsoleGame
- Logika
  - Apple
  - BodyPart
  - Snake

Ovdje su GUI i Konzola interface package-i, dok je u package-u Logika smještena sva logika igrice, kao što i samo ime označava.

Logika

U ovom dijelu, kao što je već navedeno, nalaze se tri klase.

Klasa ***Apple*** opisuje dozvoljenu prepreku, tj. ono što zmija treba da pokupi. Ova klasa sadrži osnovne informacije, atribute *xCoor*, *yCoor*, *width*, *height,* zatim *konstruktor* koji postavlja sve ove vrijednosti na zadane, tj. one koje su proslijeđene kada se inicijalizira objekat ove klase. Sadrži još funkciju *draw(Graphics g)(...)* koju koristimo da bi iscrtali objekat, *getters* i *setters* za *xCoor* i *yCoor.*

Klasa ***BodyPart*** opisuje jednu dužinu zmije, tj. zmija se sastoji od objekata ove klase, onoliko njih kolika je dužina zmije. Isto kao klasa Apple, i ova klasa sadrži atribute *xCoor*, *yCoor*, *width*, *height, konstruktor* koji postavlja sve ove vrijednosti na zadane, funkciju *draw(Graphics g)(...)* koju koristimo da bi isrtali objekat, *getters* i *setters* za *xCoor* i *yCoor.*

Klasa ***Snake*** je glavna klasa, jer je u njoj definisana sva logika za kreiranje zmije u toku igre. Sadrži atribute *running, paused, xCoor, yCoor, size, ticks. Running* označava da li se zmija kreće, *paused* da li je zmija pauzirana, *size* je početna veličina zmije, *ticks* je atribut koji koristi GUI dio projekata, da bi mogao pomjerati zmiju po java Frame prozoru.
Atributi *BodyPart b, ArrayList<BodyPart> snake, Apple apple, ArrayList<Apple> apples*; *b* je pomoćni atribut koji koristimo da bi u niz snake dodali *BodyPart*, isto tako *apple* da bi dodali *Apple* u niz *apples. Random r* koristimo za dodavanje *apple* na random poziciju. Konstruktor inicijalizira atribute *r, snake i apples* i pozove funkciju *start().* Funkcija *start()* postavi osnovne atribute na početne vrijednosti, očisti niz *snake i apples,* što na početku i nema potreba,

ali koristi prilikom restartovanja igrice. Ostale funkcije koriste package-i GUI i Konzola.
Funkciju *tick_game()* koristi GUI, a ona provjerava (ako zmija nije pauzirana) da li je niz *snake* prazan, ako jeste dodaje mu jedan element, zatim isto provjerava za niz *apples,* i dodaje element ukoliko je niz prazan. Zatim, ukoliko se *xCoor i yCoor (*nove koordinate zmije) poklope sa *xCoor i yCoor* od *apples* dužina zmije se povećava, i potrebno je ukloniti apple sa tog mjesta. Dalje se provjerava da li su nove koordinate zmije, *xCoor i yCoor*, nedozvoljene koordinate, tj. da li je zmija udarila u zid, ako jesu igra se zaustavlja pozivom funkcije *stop()* iz klase *Game*.
if(ticks > 250000){...} provjerava da li treba promijeniti koordinate, ako da doda novi element nizu *snake*, i ako ne treba povećati dužinu ovog niza ukloni mu prvi element, što ustvari predstavlja zadnji dio tijela zmije.
Funkcije *fieldInit(), snakeInit(Snake s), int[] setApple(int w, int h), stop_c()* koristi Konzola package. *fieldInit()* postavlja polje igre ako igrač igra u konzoli. *snakeInit(Snake s)* dodaje nizu *s.snake* nove elemente, sve dok veličina ne bude jednaka *size* i još, pošto se koristi u konzoli, stavlja polja *fielda* čije koordinate odgovaraju koordinatama *s.snake* tako da one sadrže zmiju. *int[] setApple(int w, int h)* postavlja vraća random kooridinatu x i y, koje se nalaze unutar polja *field* i nisu već zauzete, tj. ne odgovaraju kooridnatama x i y u nizu *snake. stop_c()* zaustavlja igru.

GUI

U ovom dijelu nalaze se dvije klase.

U klasi ***Frame*** postavlja se glavni okvir u kojem se igra, i ona ima osnovne komande, od kojih pored pozicije okvira, načina zatvaranja dodaje objekat klase *Game* u kojoj su definisani pokreti igrača, tj. interakcija sa tastaturom.

U klasi ***Game*** nalaze se postavke igre. Tu su definisani *WIDTH, HEIGHT* atributi, koji su već na početku inicijalizirani i označavaju širinu i visinu prozora u kojem se igra. Atributi *right, left, up, down* su tipa boolean i sadrže informacije o kretanju zmije. *Snake s* je objekat klase *Snake* i sa njim je povezujemo logiku igre. *thread* je objekat kalse *Thread* koji omogućava da se zmija kreće po polju, runnable dio. *start* i *br* koristimo da bi lakše organizovali prikaz poruka igraču. Dalje imamo funkciju stop(), koju pozivamo kada je kraj igre, bilo da je korisnik resetovao igru ili da je izgubio. U ovoj funkciji se kontroliše *thread*, da prilikom ponovnog pokretanja igre kada se u funkciji *start()* pokrene novi *thread* ne bi bilo prethodno nezaustavljenih.
Funkcija *paint()* iscrtava sve potrebne komponente na ekranu. *run()* je funkcija koja inače pokreće thread, pripada klasi *Runnable* i u ovom projektu postavljena je da dok god nije kraj igre poziva funkciju *tick_game()* iz klase *Snake*, i iscrta nove postavke ekrana. Pošto se ova funkcija poziva stalno dobijemo efekat pomjerajuće zmije na ekranu.
U funkciji *keyPressed(KeyEvent e)* definisani su događaji na određene tipke: promjene potrebnih atributa, pozivi potrebnih funkcija.

Konzola

Package Konzola ima dvije klase*.*

Klasa ***ConsoleGame*** ima atribute: *WIDTH, HEIGHT* čuvaju informaciju o dimenzijama polja u kojem se igra; *right, left, up, down* su tipa boolean i njihova vrijednost je true samo kada se zmija kreće u tom smjeru; *stop* označava da li je kraj igre, tj. korisnik će moći da unosi komande sve dok je *stop* false; *filed* je polje igre, *s* zmija u polju, *apple* je dozvoljena prepreka.
Konstruktor inicijalizira objekat *s* i pozove funkciju *start()*. Dalje, funkcija *start()* ispiše igraču poruku sa pravilima igre, postavi atribute na početne vrijednosti, poziva funkciju *paint()* koja iscrta elemente od *field* niza koji ustvari opisuju izgled polja igre, i pozove funkciju *play()*. U ovoj funkciji, sve dok nije kraj igre, poziva se funkcija *getCommand()* iz klase *Console* i rezultat se provjerava, tj. provjerava se vrijednost atributa move iz klase *Console* i u zavisnosti od njega dalje se izvršavaju potrebne komande. Nakon provjere komande, poziva se funkcija *move()* u kojoj se vrši izmjena stanja na polja igre, poput pomjeranja zmije, postavke nove prepreke i slično. Nakon toga pozivom funkcije *paint()* iscrta se novo stanje polja i ponavlja se postupak *play()* funkcije.

Klasa ***Console*** je klasa u kojoj započinje igra snake. Ona sadrži atribute: *move* čuva sljedeću komandu, *prevMove* čuva prethodnu ispravnu komandu - ako je korisnik unijeo neku komandu koja nije definisana onda je *move = prevMove,* i objakat klase *ConsoleGame gc*. Svi ovi atributi se inicijaliziraju u konstruktoru čime se postavi polje igre i čeka se unos korisnika. U funkciji *getCommand()*, koja se poziva iz klase ConsoleGame, postavlja se vrijednost *move* na unesenu vrijednost.