SoulEater

Specifikacija projekta

Verzija 1.0

Studentski tim: Marko Banovac

Matej Đanić

Kristijan Vrbanc

Tibor Žukina

Predmet: Virtualna Okruženja

Sadržaj

1. Objekti 3

2. Parametri i varijable 4

3. Ekrani 5

4. Poslužiteljske akcije 7

5. Baza podataka 8

6. Literatura 9

Specifikacija projekta

# Objekti

Pomični objekti se generiraju u vertikalno grupiranim skupovima (stupcima) na desnoj strani ekrana i putuju prema demonu koji se nalazi s krajnje lijeve strane ekrana, u svakom skupu postoji točno određeni broj objekata.

* **Duša** - generira se na određenoj visini na ekranu kao jedan od objekata u vertikalno usmjerenom skupu, u vertikalnom skupu generiranih objekata mora biti barem jedna duša, u slučaju dodira s demonom on ju uhvati i usisa u mračni portal iza njega koji vodi u pakao, nema nikakve dodatne atribute
* **Anđeo** - generira se na određenoj visini na ekranu kao jedan od objekata u vertikalno usmjerenom skupu, ima atribut štete koji se nasumično određuje u određenom intervalu pozitivnih cijelih brojeva, u slučaju dodira s demonom, energija se smanjuje za iznos koji je jednak atributu štete anđela
* **Tamna energija** - generira se na određenoj visini na ekranu kao jedan od objekata u vertikalno usmjerenom skupu, u slučaju dodira s demonom, energija se povećava za jedan, osim ako već nije maksimalna
* **Štit** - generira se na određenoj visini na ekranu kao jedan od objekata u vertikalno usmjerenom skupu, ima atribut trajanja zaštite u sekundama koji se nasumično određuje u određenom intervalu pozitivnih cijelih brojeva, u slučaju dodira s demonom, oko demona stvara vatreni obruč koji će ga tijekom trajanja zaštite učiniti imunim na sudare s anđelima
* **Demon** - smješten je na lijevom rubu ekrana i ima konstantan horizontalni položaj, a igrač ga može pomicati vertikalno, predstavljen je klasom koja ima samo jednu instancu objekta, ima atribut energije koja kreće od određenog maksimalnog iznosa, smanjuje se u dodiru s anđelima, a povećava u dodiru s tamnom energijom

# Parametri i varijable

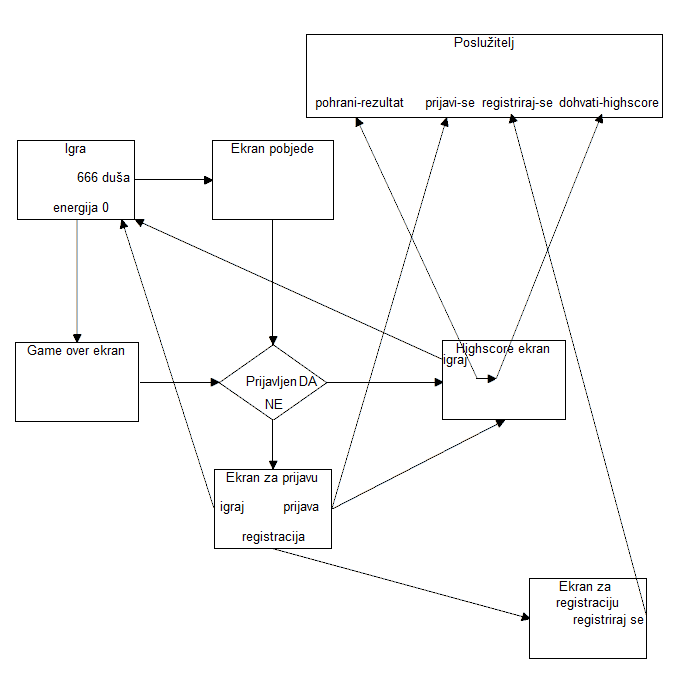
Parametri i varijable unutar igre prikazuju se igraču na ekranu, koriste se za naknadno generiranje tablice rezultata (*highscore*) te pojedinim uvjetima određuju ishod igre, kao što je poraz, pobjeda ili prijelaz na iduću razinu.

* **Broj skupljenih duša** - broj duša koje je demon sakupio tijekom igre, počinje od nule, a kada dostigne 666, igra završava pobjedom
* **Razina igre** - razina igre direktno ovisi o broju skupljenih duša, kreće od razine 1 i povećava se nakon svakih 50 duša, dakle dostignuta razina jednoznačno je određena brojem trenutno skupljenih duša te se računa po formuli ceil(brojDuša/50), igra ima 14 razina te posljednja razina završava kad je sakupljeno 666 duša, svaka razina ima drugačiju pozadinu, a indeks pozadine koja se koristi određen je formulom razina%brojPozadina, svakim prelaskom na višu razinu brzina toka objekata prema demonu se povećava
* **Energija demona** - može se promatrati kao parametar igre ili kao atribut demona, iznos može biti između maksimalnog iznosa i nule, a kad jednom dosegne nulu, igra završava porazom

# Ekrani

Ekrani unutar igre i njihov tok određeni su pravilima i ishodom igre te ponašanjem igrača.

* **Igra** - glavni ekran unutar kojeg se odvija igra, ako energija dosegne 0 prelazi na *gameover* ekran, a ako broj skupljenih duša dosegne 666 prelazi na ekran pobjede
* ***GameOver* ekran** - prikazuje se nakon poraza igrača, uz prikaz broja duša koje je dostigao, sliku skupine anđela i zvuk zlonamjernog smijeha, nakon tog ekrana prelazi se na *highscore* ekran ako je igrač prijavljen ili na ekran za prijavu ako nije
* **Ekran pobjede** - prikazuje se nakon pobjede igrača, uz prikaz odgovarajuće poruke o pobjedi, sliku skupine demona i pobjedničku glazbu, nakon tog ekrana prelazi se na *highscore* ekran ako je igrač prijavljen ili na ekran za prijavu ako nije
* **Ekran za prijavu** - prikazuje se kako bi se igrač prijavio nakon završene igre svojim korisničkim imenom i lozinkom u svrhu slanja rezultata na poslužitelj i prikaz *highscorea*, s njega je moguće otići na ekran za registraciju, prijaviti se i time otići na *highscore* ekran ili započeti ponovnu igru bez prijave
* **Ekran za registraciju** - prikazuje se kako bi se igrač registrirao ako odabere tu opciju na ekranu za prijavu, na tom ekranu igrač unosi korisničko ime, lozinku i potvrdu lozinke te se uspješno registrira ukoliko korisničko ime već ne postoji
* ***Highscore* ekran** - nakon dolaska na taj ekran najprije se poslužitelju šalje rezultat korisnika kojeg je prethodno ostvario, a potom se sa poslužitelja dohvaća i prikazuje *highscore* određenog broja najboljih korisnika, uključujući i ostvareni rezultat igrača ako se on nalazi među najboljima, s *highscore* ekrana moguća je opcija ponovne igre, odnosno povratka na glavni ekran igre u kojem ona počine ispočetka



Slika 1 Dijagram toka aktivnosti

# Poslužiteljske akcije

Iako je moguća igra u *singleplayer* načinu bez prijave i pohrane te spremanja rezultata, poslužitelj pruža i obrađuje zahtjeve za slanje i pohranu rezultata i uvid u rezultate drugih igrača[[1]](#footnote-1).

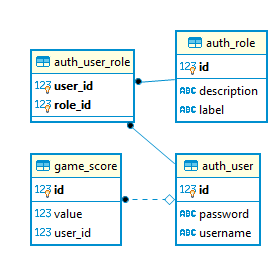
* ***Register*** – poslužitelju se šalje zahtjev koji sadrži kao parametre korisničko ime i lozinku, a kao rezultat poruku da je registracija (ne)uspješna (ukoliko već postoji korisnik s tim korisničkim imenom). Ovaj zahtjev rezultira generiranjem zapisa o korisniku u bazi podataka (userId, username, password).
* ***Login*** – poslužitelju se šalje zahtjev koji sadrži parametre korisničko ime i lozinku, potom generira i vraća jedinstveni autorizacijski token koji se pohranjuje na uređaj korisnika ako je prijava uspješna ili poruku da je prijava neuspješna (ukoliko ne postoji korisnik s tim korisničkim imenom ili je upisana lozinka pogrešna). Autorizacijski token kojeg generira ovaj zahtjev se koristi u svim sljedećim zahtjevima za upisivanje rezultata kako bi se osigurao autorizirani pristup bazi podataka
* ***AddScore*** – zahtjev koji se šalje krajem svake odigrane igre, a kao parametar sadrži ostvareni rezultat u obliku broja sakupljenih duša tijekom igre, a vraća informaciju o uspješnosti pohrane rezultata. Ovaj zahtjev rezultira generiranjem ili ažuriranjem zapisa o najboljem postignutom rezultatu pojedinog korisnika. Rezultat se generira ako korisnik dosad nije ostvario rezultat, tj. ne postoji zapis rezultata trenutnog korisnika u bazi podataka, a ažurira se samo ako je postignuti rezultat veći od postojećeg.
* ***GetScores*** – Zahtjev koji dohvaća sortiranu listu svih postignutih rezultata i korisničkih imena korisnika koji su ih ostvarili. Lista je sortirana po vrijednosti sakupljenih duša od najvećeg prema najmanjem. Ovaj zahtjev ne generira zapise u bazi podataka već ih samo iščitava.

Primjer tijela (eng. *„Request body“*) zahjeva koji se šalju putem ***Login/Request*** akcija prikazani su ispod:

|  |
| --- |
| {  “username”: “John Doe”,  “password”: “password123”  } |

# Baza podataka

Baza podataka sadrži objekte o registriranim korisnicima, njihovim ulogama i ostvarenim rezultatima. Slika 1 prikazuje *Entity-Relation* dijagram korištene baze podataka. Baza podataka implementirana kao PostgreSQL baza.



Slika 2 *ER* dijagram baze podataka

* **Korisnik** (*„auth\_user“*)– zapis registriranog korisnika
  + ***id*** (AUTO\_INCREMENT, PRIMARY KEY)
  + ***username*** – jedinstveno korisničko ime
  + ***password*** – generirani sažetak, tj. *hash* lozinke
* **Uloga** (*„auth\_role“*)– zapis o ulozi pojedinog korisnika prema kojoj se određuju točno koje privilegije i ovlasti manipulacije nad podatcima korisnik posjeduje
  + ***id*** (AUTO\_INCREMENT, PRIMARY KEY)
  + ***label*** – naziv uloge
  + ***description*** – opis uloge
* **Rezultat** *(„game\_score“*) – zapis najvećeg postignutog rezultata za pojedinog korisnika
  + ***id*** (AUTO\_INCREMENT, PRIMARY KEY)
  + ***user\_id*** – strani ključ na tablicu korisnika
  + ***value*** – vrijednost ostvarenog rezultata
* ***Korisnik-Uloga*** (*„auth\_user\_role“*) – spojna (eng. *„junction“*) tablica koja sadrži zapise o posjedovanju uloga za svakog korisnika. Kao zadana (početna) vrijednost, svaki novoregistrirani korisnik posjeduje ulogu *„User“*
  + ***user\_id*** –strani ključ na tablicu korisnika
  + ***role\_id*** –strani ključ na tablicu korisnika

# Literatura

1. <https://devcenter.heroku.com/categories/reference> - *Heroku* platforma
2. <https://swagger.io/docs/> - Generirana *API* dokumentacija
3. <https://docs.spring.io/spring-framework/docs/current/reference/html/> - *Spring* radni okvir
4. <https://www.postgresql.org/docs/> - PostgreSQL baza podataka
5. <https://www.baeldung.com/learn-spring-security-course?utm_source=github&utm_medium=social&utm_content=general&utm_campaign=lss> – Implementacija autorizacije (registracija, prijava…)
6. <https://spring.io/guides/tutorials/rest/> - *Rest* arhitektura web servisa

1. Generirani ***API*** moguće je pronaći na sljedećoj poveznici: <https://virokr.herokuapp.com/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config> [↑](#footnote-ref-1)