Vaje 2019/2020 Razvoj računalniških iger

20. december 2019

1 Moja računalniška igra

Po vzoru igre iz predavanj (reševanje astronautov ver. 01) si izmisli/zamisli svojo igro. Igro boš izpopolnjeval in spreminjal skozi več vaj, vendar sama ideja ni bistvena za oceno.

Pri izbiri igre nimate čisto prostih rok! Igra naj vsebuje podobne koncepte kot primer!

1.1 Idejna zasnova

Osnovni koncept igre naj bo zelo podoben igri, ki smo jo pripravili za zgled. Igra naj ne vsebuje preveč različnih elementov. Igra bo namenjena učenju na primeru. Vsebuje naj samo eno stopnjo, brez intro zaslona,... Igro lahko kasneje spremenite, ...

Ideje:

- Premika se lahko od leve proti desni, desna proti levi.
- Ovire so lahko različne (ne preveč ovir, npr. dva različna tipa).
- Pobira se lahko cekine, gorivo, metke, hrano, ...
- Premikanje ovir ni nujno da je samo navzdol.
- Če planirate odboj, ni nujno da to implementirate sedaj, saj boste to lahko dodali kasneje.

1.2 Osnutek načrta

Napiši predsatvitev (google doc), kjer boš na kratko opisal:

- Skiciraj na papir (opcijsko).
- Dinamiko igre.
- Mehaniko igre.
- Elemente igre.

1.3 Implementacija Lecture01 - obvezna pred 02 implementacijo

Implementiraj svojo igro in se drži naslednjih pravil:

- Implementacija naj bo blizu podane.
- Uporabljaj podobna poimenovanja.
- Vsebuje naj vse koncepte iz primera (Array, Rectangle, deltaTime, overlaps, sound, basic font, ...).
- Vse dimenzije naj bodo v pikslih (1:1), kasneje bomo spoznali druge metode.

2 Implementacja 02 - po drugem predavanju

Naredite kopijo igre 01. Po vzoru iz predavanj, ponovno implementirajte igro, vendar tokrat naj bo objektno orientirana implementacija. Vsak element igre naj bo svoj objekt in vsak tip svoj razred. Pomagaj si z razredi GameObject, GameObjectDynamic, ... Pomagaj si s primerom iz spleta: super jumper

Igra naj vsebuje:

- Razrede, dedovanja, ...
- Individualno obnašanje (rotacija, različne hitrosti, velikosti,...)

2.1 Mini preizkus - pospešek in hitrost (NEOBVEZNA)

Naredi simulacijo žogice, ki se odbija. Preizkusi različne pospeške. Primer:

- Tam kjer klikneš z miško se pojavi žogica.
- Žogica se začne premikat navzdol, ko se dotakne dna se odbije. pri tem zgubi del hitrosti.
- Odboj žoge

2.2 Mini preizkus - rotacija, hitrost (NEOBVEZNA)

Simuliraj kolo, ki se vrti od leve proti desni strani zaslona. Kolo naj ima špice, da se bolje vidi učinek.

3 Preverjanje konceptov

Izmisli si primer (lahko na svojem projektu), kjer demonstriraš:

- Premikanje in povečave scene s pomočjo kamere.
- Uporaba različnega pogleda (viewport).
- Uporaba dimenzij sveta.
- Uporaba draw metode, tako da se uporabi scale, rotate...

3.1 Pouporaba objektov

V lastni igri (Vaja Implementacija 02) implementiraj pouporabo objektov (pooling) po zgledu iz predavanj.

4 Razhroščevanje in efekti sistemov delcev

V lastni igri dodaj možnosti za razhroščevanje po vzoru iz predavanj (npr. ob pritisku F5 tipke).

- Izris mreže.
- Izriši meje okoli vseh elementov igre.
- Premikanje kamere, povečave,...
- Izpis uporabljenega pomnilnika.

4.1 Efekti sistemov delcev

Izdelaj in lastni igri smiselno dodaj efekte sistemov delcev.

5 Namizna igra

Poišči/izberi eno preprosto namizno igro (board game), ter jo implementiraj. Implementacija bo potekala tekom več vaj.

Isti tip igre lahko izbere več študentov, vendar ima vsak svojo implementacijo. Vendar ne več kot 5 študentov na tip igre.

Projekt oddaj na github.

5.1 Osnutek načrta

Napiši predsatvitev (readme datoteka na github), kjer boš na kratko opisal:

- Izgled oz. sele skiciraj na papir (opcijsko).
- Dinamiko igre.
- Mehaniko igre.
- Elemente igre.

5.2 Logika in stanje igre

Stanje igre opiši s pomočjo tabel in bitnih zastavic ali EnumSet razreda. Naredi po vzoru igre Minolovec iz predavanj.

6 Namizna igra 2. del

Cilj te naloge je nadaljevanje razvoja namizne igre (pokažeš napredek od prejšnje naloge plus dodatne funkcionalnosti). V igri uporabi Assetmanager, Atlas, demonstriraj načrtovalski vzorec Descriptor, po vzoru iz predavanj.

6.1 Animacija

Igro nadgradi z vsaj eno lastno diskretno animacijo.

6.2 Zvočni efekt

Igro nadgradi z vsaj enim lastnim (generiranim) zvočnim efektom.

6.3 Opomba

Zaradi velikih razlik pri kompleksnosti izbranih iger, se pričakuje da so lažje igre bolj dodelane. Za minimalno oceno morajo igre vsebovati vse elemente opisane v navodilih naloge.

7 Namizna igra 3. del

Do sedaj ste se osredotočili na gameplay igre. Namizni igri je potrebno dodati naslednje elemente.

- Igra naj vsebuje razred GameManager, ki skrbi za shranjevanje rezultatov in druga stanja (nastavitve, ime igralca, ...) ki so pomembna za pred in po igri.
- Začetni ekran/okno, ki vsebuje funkcionalnosti kot so: začni novo igro, lestvica oz rezultati, izbereš zahtevnost, mogoče vklop izklop zvoka, ...
- Ekran/okno lestvica/rezultati kjer so podatki o rezultatih.

* Vsaj pri ekranu lestvice uporabite LibGDX gradnike Scene2d, priporočamo tudi pri začetnem ekranu. V primeru, da omenjeni ekrani niso smiselni za vašo igro, jih lahko nadomestite z drugimi ali pa naredite nov projekt kjer demonstrirate uporabo.

8 Implementacija ECS

Cilj naloge je, da se na praktičnem primeru spoznate z arhitekturnim načrtovalskim vzorcem ECS. V ta namen na novo implementirajte igro iz prvega sklopa vaj (moja prva igra). Za zgled vzemite primer iz predavanj (Astronaut ECS).

8.1 Definiranje komponent, entitet in sistemov

V okviru te vaje naredite spletno razpredelnico, kjer definirate komponente, entitete, sisteme in njihove relacije za svojo igro. Za vzor uporabite razpredelnico Astronaut ECS relacije. Povezavo do vaše razpredelnice zapišite v razpredelnico podano na strani učne enote (*Vaja 8.1: Razpredelnica*).

Priporočljivo je, da to nalogo zagovorite oz. vsaj pokažete na vajah preden se lotite implementacije v drugem delu naloge. Drugače se lahko zgodi, da si boste nalogo narobe zastavili (narobe definirali komponente, entitete, sisteme) in boste pri implementaciji imeli velike težave.

8.2 Implementacija

Implementacija igre s pomočjo arhitekturnega načrtovalskega vzorca ECS. Projekt naj vsebuje podporo za Android.

Nasveti:

- Projekt začnite od začetka (pri generiranju projekta izberite *Desktop*, *Android*, *Tools* in *Ashley*).
- Skopirajte vire (assets) iz starega projekta.
- Ustvarite primerno podatkovno strukturo (pakete).
- Skopirajte in ustrezno popravite AssetManager, GameManager, Descriptor,...
- Skopirajte komponente in sisteme, ki mislite da so enaki primeru iz predavanj, ter dodate nove.
- Skopirajte in ustrezno poimenujte razrede GameX in GameScreen.
- Popravite in dopišite ostalo kodo.

Bodite pozorni da bodo vsi razredi in spremenljivke smiselno poimenovani glede na vašo igro (pri študentih pogosto opazimo, da dele kode pridobljene iz primerov niti ne preimenujejo).

9 Izdelava stopenj s pomočjo urejevalnika TiledMap

Za osnovo vzemite primer iz predavanj in:

- Tematsko spremenite/preoblikujte igro in uporabite drugi nabor ploščic (iz interneta). Uporabite orodje Tiled.
- Demonstrirajte ovire tako da uporabite object layer.
- * Za višjo oceno implementirajte območja (uporabite dodaten object layer), kjer se spremeni barva/videz glavnega akterja (igralca). Kot primer bi si lahko zamislili da v srednjeveški igri, ko vitez pride v območje močvirja spremeni videz in se začne premikati počasneje. Izven območja se ponovno spremeni.

10 Uporba Box2D

Cilj naloge je da raziščete možne uporabe knjižnice Box2D. Predlagamo, da si pogledate kakšno orodje, video vadnico (youtube ;)), spletno vadnico (blog, ...). Da lahko preverimo ali ste nalogo opravili zadovoljivo podajte kratko poročilo, ki vsebuje:

- Povezavo na vir ali vire.
- Kratek opis cilja in rešitve.
- Seznam vseh Box2D elementov oz. konceptov, ki so uporabljeni.

11 LibGDX Box2D (neobvezna)

Igro iz poglavija TiledMap (ali kaj svojega; poleg TiledMap naj uporablja še arhitekturni vzorec ECS) spremeni tako, da bo uporabljala Box2D. Uporabite ga za:

- Omejitev sveta (zidovi, ovire, ...) vnesi v fizikalen svet.
- Pobiranje implementiraj s pomočjo sistema trkov v Box2D.
- Dodaj premikajoče se elemente. Npr. s pomočjo kosilnice odrinem žogo med košenjem trave.