Snowman slider
Tehnička dokumentacija
Verzija 1.0

Student: Mateja Golec, 0036533847

Sadržaj

Opis razvijenog proizvoda

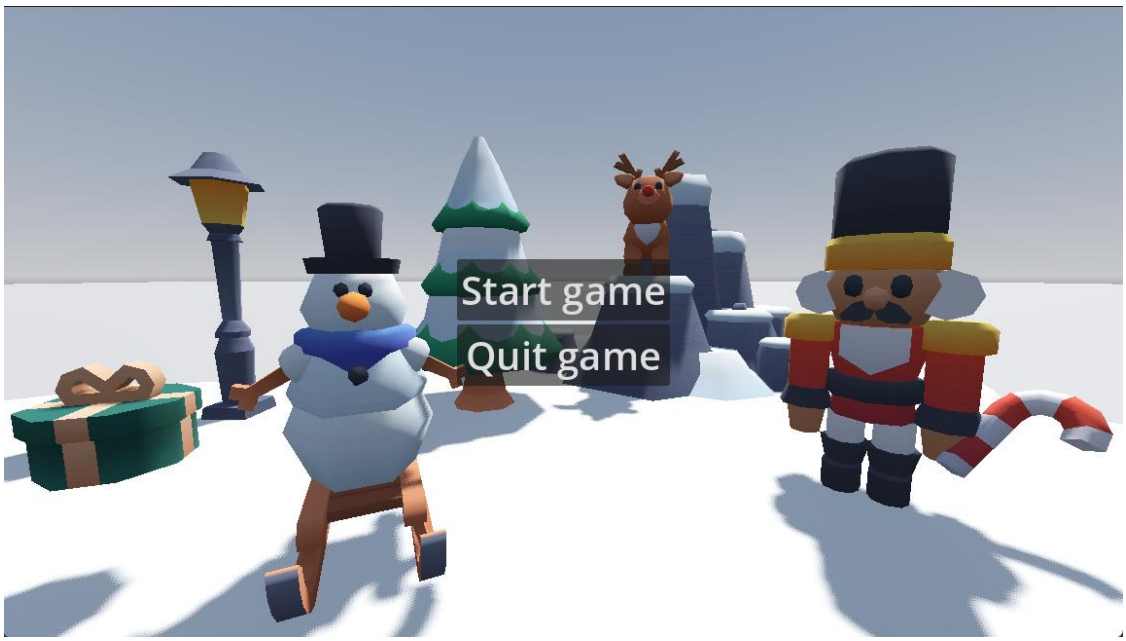
U sklopu treće (samostalne) laboratorijske vježbe implementirana je jednostavna 3D igrica nazvana „Snowman slider“.

Igrica je primjer tzv. „endless runner“ računalne igre – igre u kojoj igrač kontrolira lik koji se prividno neprestano kreće unaprijed kroz beskonačni svijet.

Pri tome kretanju cilj igre je izbjeći prepreke – udarac u prepreku označava kraj same igre. Posljedični cilj je preživjeti što duže igrajući samu igru te usput sakupiti što više poklona. Završna scena prikazuje koji je uspjeh igrača – vrijeme izdržano igrajući te broj sakupljenih poklona.

Odmicanjem vremena povećava se brzina kretanja što otežava izbjegavanje prepreka terena koje se stvaraju slučajnim odabirom.

Sam cilj izrade vježbe bio je upoznati se novom tehnologijom za izradu računalnih igara te shvatiti osnovne koncepte i način funkcioniranja kako stvarati svijet igrice i kako naučeno znanje na predmetu iskoristiti na praktičnom primjeru.



Slika 1 Početni ekran igrice

1. Tehničke značajke

Igrica je razvijena koristeći Godot, besplatan i open-source alat za izradu računalnih igara.

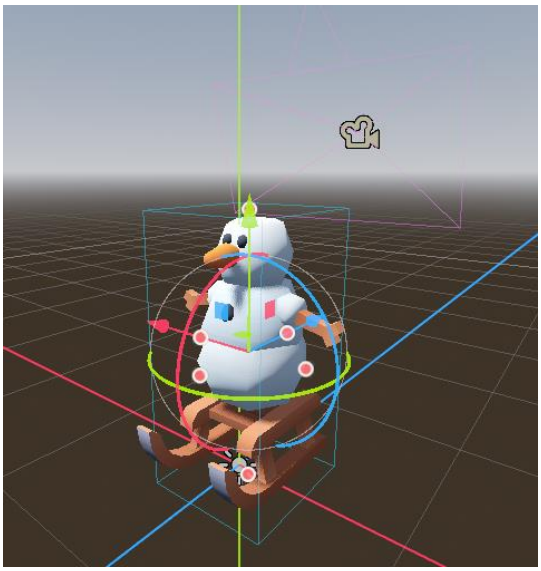
Igrica se sastoji od nekoliko ključnih dijelova:

1. Igrač (Player)

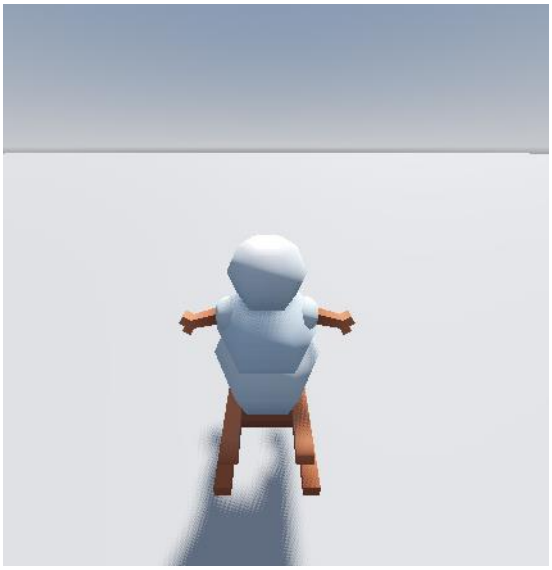
Igra je dizajnirana iz perspektive trećeg lica – igrač vidi lik koji se kliže kroz scenu i izbjegava prepreke.
Sam igrač predstavlja čvor „CharacterBody3D“ – korištenje tog čvora omogućava olakšanu kontrolu igrača u ovoj igri (predefinirana svojstva i funkcije kao npr. za brzinu, gravitaciju, kolizije s podom i ostalim objektima).
Igraču je omogućeno kretanje u smjeru x-osi koordinatnog sustava (lijevo i desno) pritiskom na tipke strelice lijevo i desno, te skakanje u smjeru y-osi korištenjem tipke razmaknice ili tipke Enter.
Sam igrač ne kreće se u sjeru z-osi (prema naprijed, već je privid tog kretanja ostvaren kretanjem terena prema igraču).
U igračevoj skripti još je definirano prepoznavanje kolizija samog čvora igrača sa čvorovima prepreka – ako se dogodi takva kolizija (sve prepreke dodane su u istu grupu – „Obstacle“) poziva se kod koji prikazuje završnu scenu.

```gdscript
extends CharacterBody3D

const SPEED = 5.0
const JUMP_VELOCITY = 4.5

func _physics_process(delta: float) -> void:

    if not is_on_floor():
        velocity += get_gravity() * delta

    if Input.is_action_just_pressed("ui_accept") and is_on_floor():
        velocity.y = JUMP_VELOCITY

    #vektor dobiven iz korisničkog unosa
    var input_dir := Input.get_vector("ui_left", "ui_right", "ui_up", "ui_down")
    #dobivanje globalnog smjera kretanja iz lokalnog 2d ulaza
    var direction := (transform.basis * Vector3(input_dir.x, 0, input_dir.y)).normalized()
    if direction:
        velocity.x = direction.x * SPEED
    else:
        velocity.x = move_toward(velocity.x, 0, SPEED) # smanjivanje brzine prema 0

    var collision = get_last_slide_collision( )
    if collision:
        var collider = collision.get_collider()
        if collider and collider.is_in_group("Obstacle"): #otkrivanje kolizije s preprekom
            end_game()
    move_and_slide()


func end_game():
    const end_scene = preload("res://scenes/endScene.tscn")
    get_tree().change_scene_to_packed(end_scene)
```

Slika 2 Skripta igrača



Slika 3 Collision shape igrača



Slika 4 Preview kamere igrača

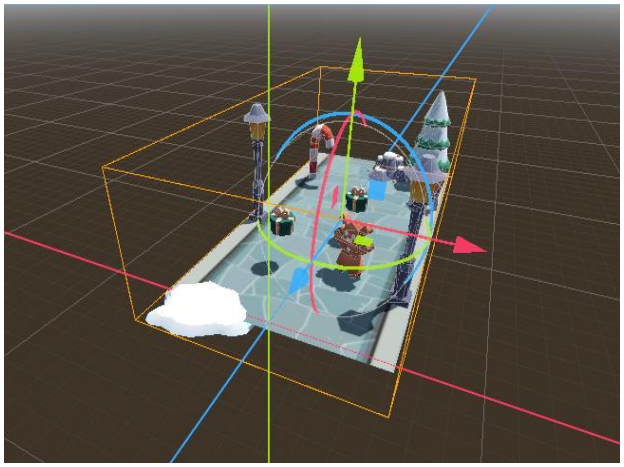
2. Tereni (Terrain)

Skripta terrain_controler.gd upravlja učitavanjem, inicijalizacijom, kretanjem i prikazom te brisanjem (ne)vidljivih terena u igrici.
Teren predstavlja plohu (pod) po kojem igrač kliže. Sa rubnih strana terena nalaze se barijere koje onemogućuju igraču kretnju izvan granica terena. Na terenu se nalaze različite prepreke koje je potrebno izbjegavati.
Implementirano je deset različitih terena od kojih se nasumično bira koji će se teren prikazati idući u sceni.
U isto vrijeme na ekranu vidljiva su 4 terena, kada se prvi teren makne iz igračevog vidljivog

područja, slučajnim odabirom učitava se novi teren te se postavlja na kraj vidljivih terena.
Brzina kretnje terena prema igraču ubrzava se s vremenom kako bi izbjegavanje prepreka postalo izazovnije.

```gdscript
extends Node
class_name TerrainController

var Terrains: Array = []
var terrain_view: Array[MeshInstance3D] = []
var min_vel = 8.5
@export var t_velocity: float = 8.5
@export var t_in_scene = 4

@export_dir var t_path = "res://terrain"
@export_dir var t_start_path = "res://terrain_start.tscn"

var start_scene: MeshInstance3D = null

func _ready() -> void:
    load_terrains(t_path) # učitavanje svih terena
    init_terrain(t_in_scene) # inicijalizacija terena za scenu


func _physics_process(delta: float) -> void:
    t_velocity = min_vel + delta * 0.1
    move_terrain(delta)
```

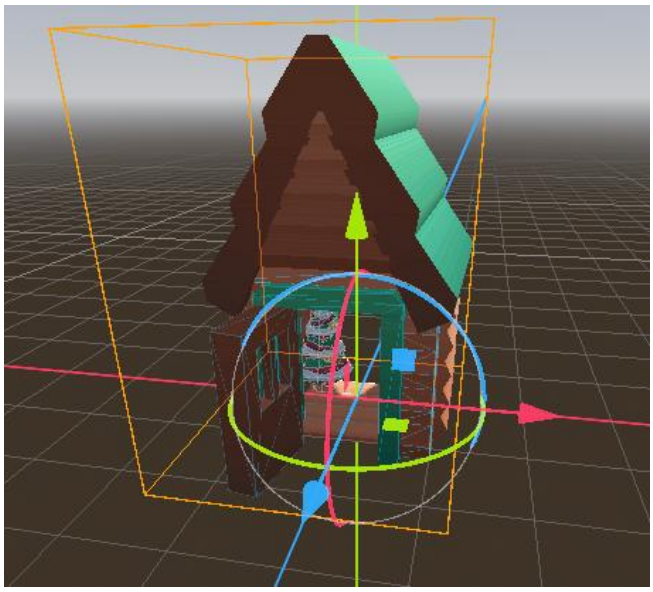
Slika 5 Isječak skripte upravljnja terenima



Slika 6 Primjer terena s objektima

3. Prepreke (Obstacle)

Prepreke predstavljaju različiti objekti preuzeti sa stranice https://www.kenney.nl/assets/holiday-kit. (Na GitHub repozitoriju dodana je licenca za slobodno korištenje dodataka).
Svakom assetu dodan je CollisionShape3D objekt te je svaki dodatak zasebno dodan u scenu. Sve prepreke su dio grupe „Obstacle“ prema čemu se prepoznaje kolizija sa igračem.



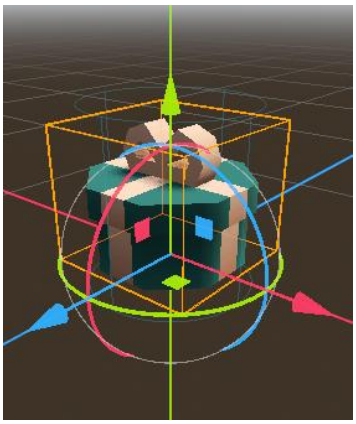
Slika 7 Primjer objekta - prepreke

4. Poklon (PickUp)

Cilj igre je uz ostvarivanje što duljeg vremena igranja, sakupiti što više poklona. Pokloni koje je potrebno sakupiti se u sceni rotiraju oko svoje ose te titraju gore dolje oko početnog položaja kako bi ih igrač mogao razlikovati od ostalih prepreka. Kada se dogodi kolizija igrača sa poklonom, poklon se uklanja sa scene te se brojač poklona uvećava.

```
1   extends Area3D
2
3   @export var rotation_speed : float = 90.0
4   @export var oscillation_amplitude : float = 0.2
5   @export var oscillation_speed : float = 5.0
6   var time_passed : float = 0.0
7
8   func _process(delta: float) -> void:
9       rotation_degrees.y += rotation_speed * delta
10      time_passed += delta
11      var new_height = oscillation_amplitude * sin(time_passed * oscillation_speed)
12      position.y = new_height + 0.5
13
14  func _on_body_entered(body: Node3D) -> void:
15      if(body.is_in_group("Player")):
16          GameManager.increase_count()
17          queue_free()
18
```

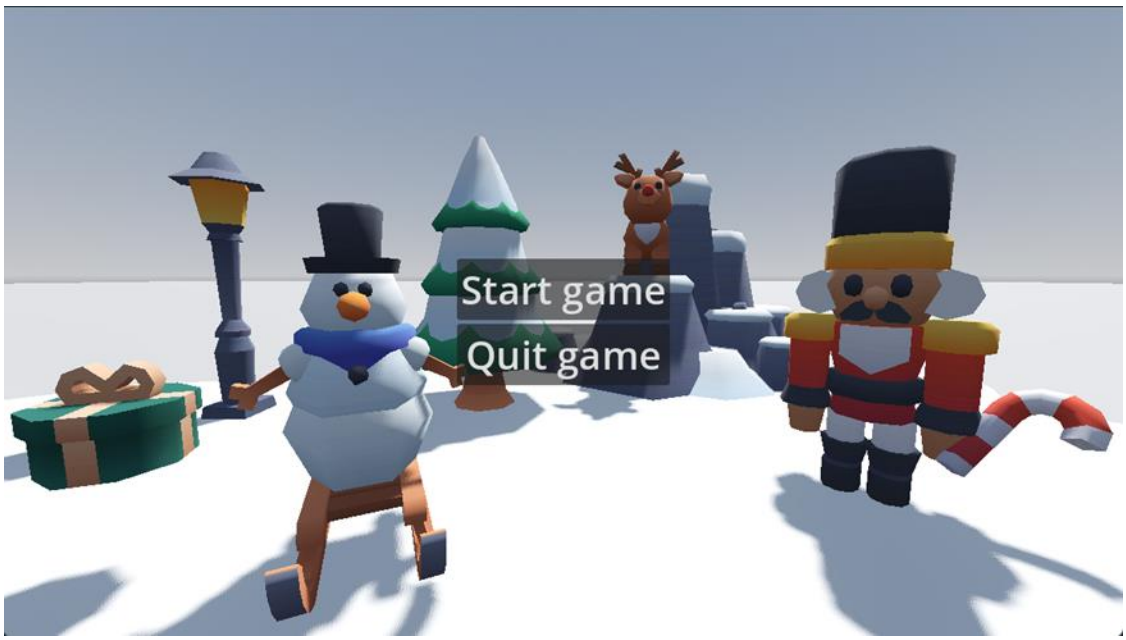
Slika 8 Skripta za upravljanje poklonom



Slika 9 Objekt poklona

5. Glavni meni

Glavni meni omogućuje početak igranja igre te izlazak iz same igre.



Slika 10 Početni ekran

6. Završni ekran

Na završnom ekranu su prikazane oznaka „Score“ – broj sakupljenih poklona i oznaka „Time“ – vrijeme igranja.
Meni na završnom ekranu omogućuje ponovo pokretanje igre ili povratak na glavni ekran.

Kontrola prebacivanja između ekrana te upravljanja varijablama odrađena je globalnom skriptom gamemanager.gd – unutar te skripte nalaze se kodovi za učitavanje i promjenu scena (prikaza na ekranu) te kodovi za mjerenje vremena i povećavanje brojača poklona prilikom kolizije sa objektom poklona.

Slika 11 Završni ekran

```gdscript
func loadGame() -> void:
	get_tree().change_scene_to_packed(GAME)
	playing = true
	timer = 0
	present_counter = 0


func loadEndScene() -> void:
	get_tree().change_scene_to_packed(END)
	playing = false

func _process(delta: float) -> void:
	if playing:
		timer += delta
		var timerLabel = get_node_or_null("/root/World/Player/Camera3D/TimerLabel")
		if timerLabel != null:
			timerLabel.text = "Time: " + str(floor(timer))

func increase_count():
	present_counter += 1
	var presentLabel = get_node_or_null("/root/World/Player/Camera3D/PickUpLabel")
	if presentLabel != null:
		presentLabel.text = str(present_counter)

```

Slika 12 Game manager skripta

2. Upute za korištenje

1. Lokalno pokretanje

1. Otvorite terminal na svom računalu te se pozicionirajte u datoteku gdje želite klonirati projekt.
2. Izvršite naredbu:

    a. git clone *https://github.com/mGolec73/racunalna_animacija.git*

3. Pokrenite Godot.
4. Kliknite na „Import“.
5. Pronađite direktorij gdje se nalazi preuzeti projekt: treci_labos/runner
6. Kliknite na Open.
7. Kliknite na Play ikonu.