KAUNAS UNIVERSITY OF TECHNOLOGY

FACULTY OF INFORMATICS

T120B166 Development of Computer Games and Interactive Applications

[3D] First Person Tower Defence

IFZm-7, Gintaras Ruočkus:

Date: *2020.10.03*

Kaunas, 2020

Tables of Contents

Description of Your Game

Description of Your Game.

1. 3D or 2D? 3D
2. What type is your game? You will have to defend your house from zombies
3. What genre is your game? First person defense game.
4. Platforms (mobile, PC or both?) PC
5. Scenario Description. In this game player has to defend his house from zombies, which can come in from 2 doors. Player will start with a pistol to defend from zombies. Zombies will give player money after the zombie dies, which can be used to buy better weapons (shotgun, rifle, etc.). Player will be able to place some blockades, which won't let zombies get through. For zombies to get trough blockades, they will have to destroy those blockades. There is going to be a possibility to upgrade blockade for it to be harder to destroy and to damage zombies. There are going to be waves, where certain amount of zombies will attack player. After all zombies dies player will be able freely put blockades with earned money. Then when player is ready, he can start another wave. The goal of this game is to survive as many round as possible.

Laboratory work #1

List of tasks

1. ***Create a small demo level using tiles, sprites or 3D objects***
2. ***Create your own game Character with own Controller script, make it move and jump.***
3. ***Decorate your World***

Solution

Task #1. *Create a small demo level using tiles, sprites or 3D objects*

For a demo level I created apartment inside rooms.

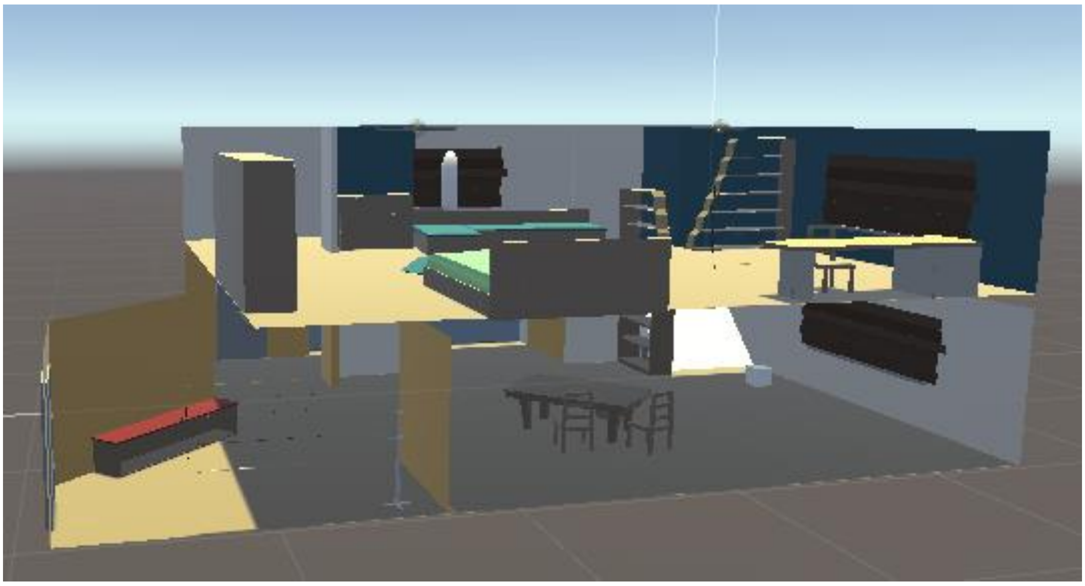


Figure 1. Apratment inside used as a map.

Task #2. *Create your own game Character with own Controller script, make it move and jump.*

As a base character has rigid body for movement. Movement is calculated by multiplying vectors right and forward with inputs respectively. Than by summing up those values and normalizing them, we get vector of player movement. For jumping I use force, which is counted by multiplying vector up with jump velocity. For camera rotation I used input which was multiplied with rotation speed.

```
void Update()
{
if(grounded == false && rb.velocity.y == 0)
     grounded = true;
PerformTransform();
if (grounded == true && Input.GetButtonDown(jump))
{
rb.AddForce(Vector3.up * jumpVelocity);
grounded = false;
}
}
```

Table 1. Movement script Update() function, calling movement function and controlling jumps.

```
private void PerformTransform()
{
// Movement count
_transform = (transform.right * Input.GetAxisRaw(mLeftRight) + transform.forward * Input.GetAxisRaw(mFrontBack)).normalized;
if (Input.GetButton(mRunning))
_velocity = _transform * runSpeed;
else
velocity = _transform * speed;

// Rotation count (y)
_rotation = new Vector3(0f, Input.GetAxisRaw(mMouseX), 0f) * rotationSpeed;

// Tilt count(x)
_tilt = Input.GetAxisRaw(mMouseY) * rotationSpeed;

if(_velocity != Vector3.zero)
    rb.MovePosition(rb.position + _velocity * Time.fixedDeltaTime);

if(_rotation != Vector3.zero)
    b.MoveRotation(rb.rotation * Quaternion.Euler(_rotation));

if(_tilt != 0)
if(cam != null)
{
Quaternion yQuaternion = Quaternion.AngleAxis(_tilt, -Vector3.right);
Quaternion temp = cam.transform.localRotation * yQuaternion;
if (Quaternion.Angle(center, temp) < maxTilt)
    cam.transform.localRotation = temp;
}
}
```

Table 2. PerformTransform() fuction, counting change in movement and trasforming player.

Task #3. *Decorate your World*

I made 18 objects using “Blender”. Than using free drawing program “Gimp” made materials. When object was ready for use, it was imported to “Unity” and placed in map. Some of the objects where used multiple times (ceiling lights, chairs, ...). For lighting I used 7 directional lights.

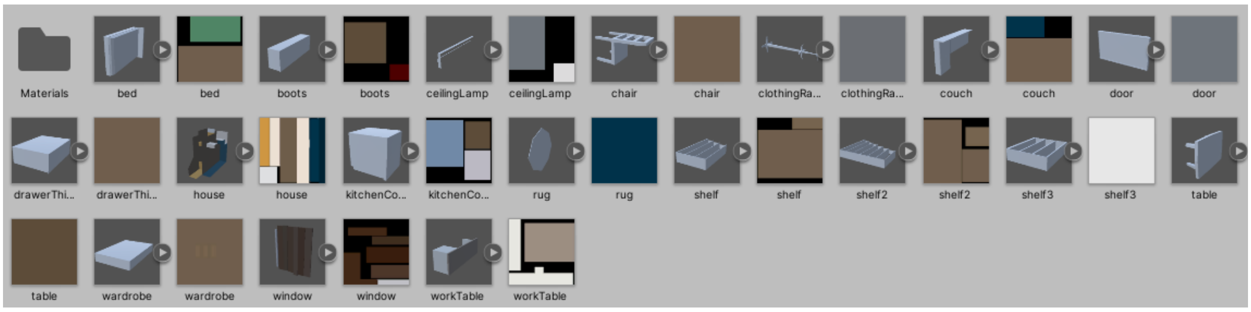


Figure 2. Created models and textures for them.

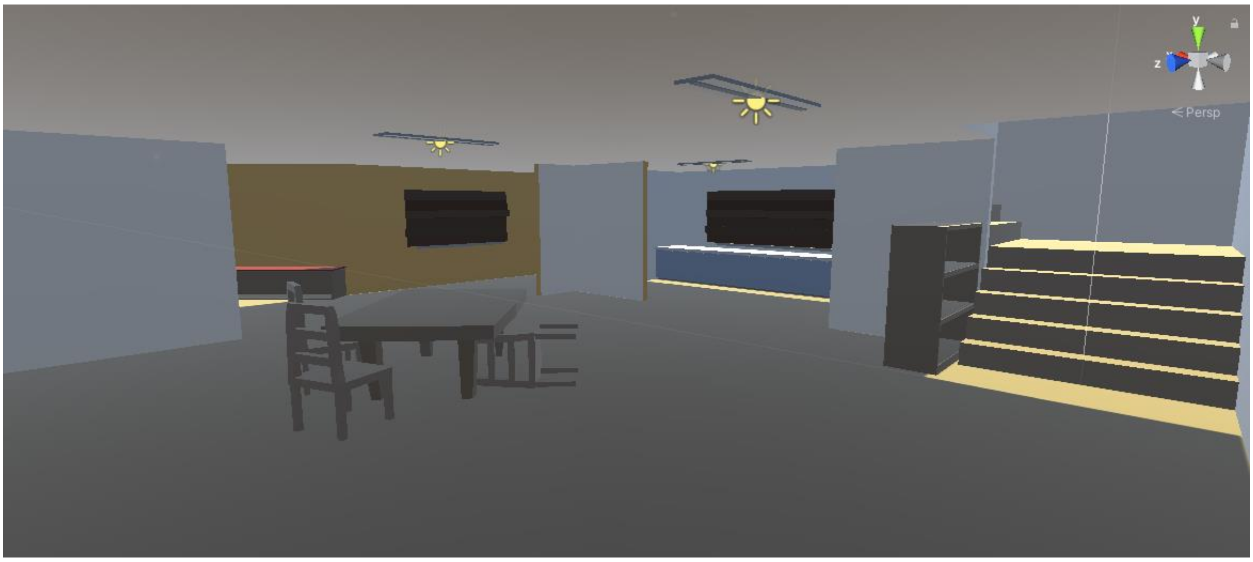


Figure 3. First floor.

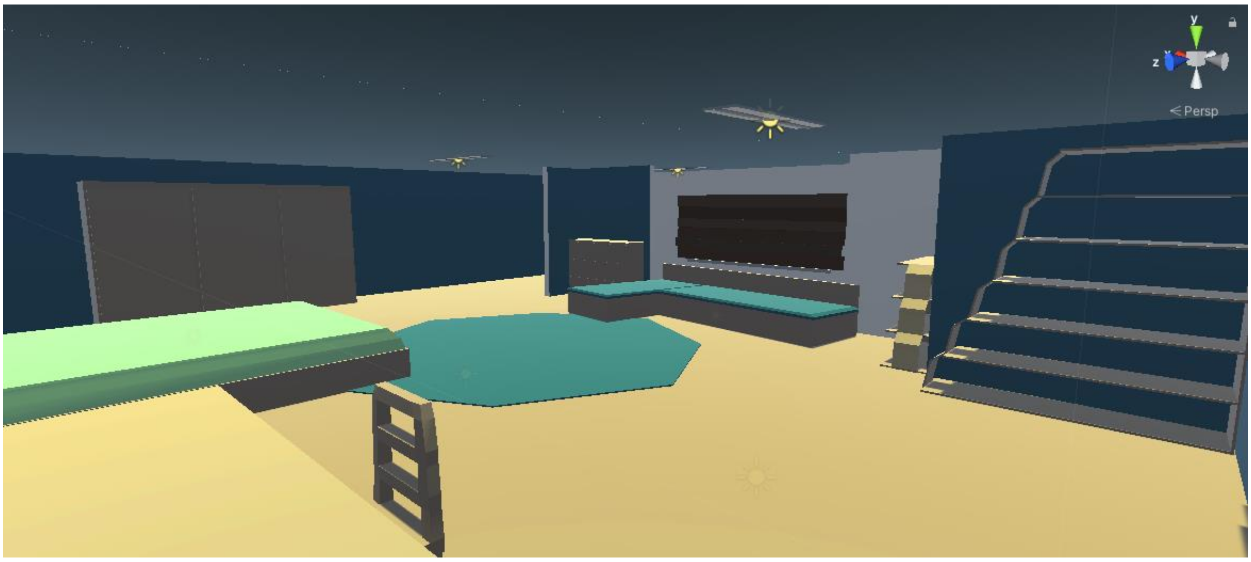


Figure 4. Second floor.

Laboratory work #2

List of tasks

1. Experiment with Polybrush / SpriteShapes / ProBuilder / Tilemap Palette tools, start expanding, building your World environment;
2. Add animations to your game character (Assuming you have a working PlayerController);
3. Create and/or animate 5 objects of your choice in your World (you can use Animator Component and/or Timeline);
4. Create at least 5 particle effects for your environment (dust, explosion, smoke gas, light sparks, etc.);
5. Add a custom skybox;
6. Create at least 3 different Physics Materials for various parts of the map (either it's a slippery platform/ice, bouncy wall, non-slippery ground, etc.);
7. Create 4 types of objects that use OnCollisionEnter / OnCollisionEnter2D / OnTriggerEnter / OnTriggerEnter2D:
–Destroys itself and spawns a particle system and adds a point;
–Plays a sound;
–Moves away using Rigidbody Force or Velocity;
–Falls down / teleports / opens up (trap, door, etc.).
8. Assign optimal colliders for the environment objects that are not moving and set their flags to static (test performance before and after);
9. Bake a lightmap and measure performance;
10. Optimize all textures depending on their parameters and measure graphical memory load;
11. Try hard vs soft shadows, different quality settings and measure the performance;

Solution

Task #1. *Experiment with Polybrush / SpriteShapes / ProBuilder / Tilemap Palette tools, start expanding, building your World environment*

With probuilder tool made few objects in the house:

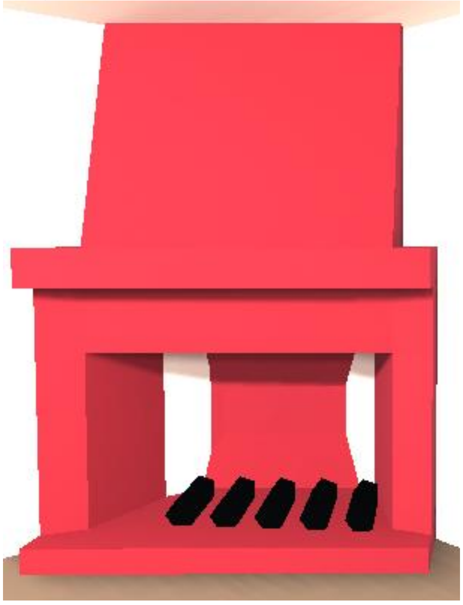


Figure 5. Fireplace (those dark things are not part of it).

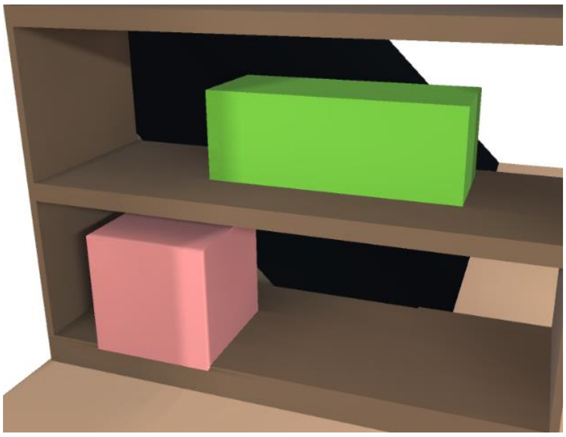


Figure 6. Two boxes.

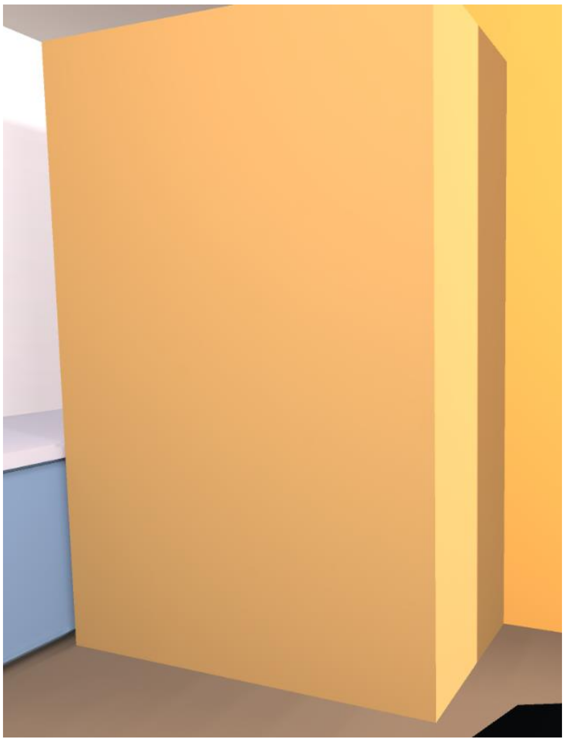


Figure 7. Fridge.

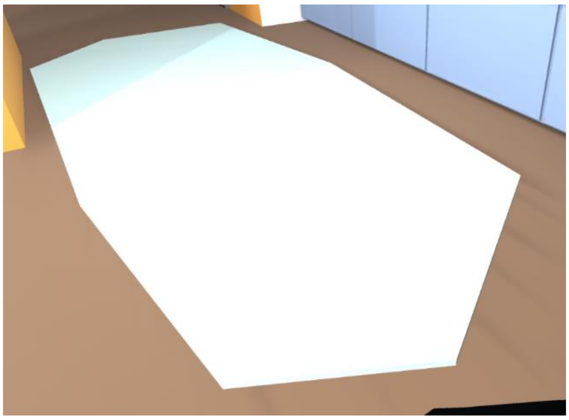


Figure 8. Slippery water.

Task #2. *Add animations to your game character (Assuming you have a working PlayerController)*

My character has 8 different weapons in comparison of their size you could differ them into 2 small weapons and 6 big weapons, which meant that I could use same animations for my character, just needed a way to attach weapon, when needed. To do that I left armature of character, because of that I was able to attach every weapon to one hand bone (accidentally got reversed names of bone – left and right should be reverse). To take another weapon I just needed to disable current weapon and go to correct animation.

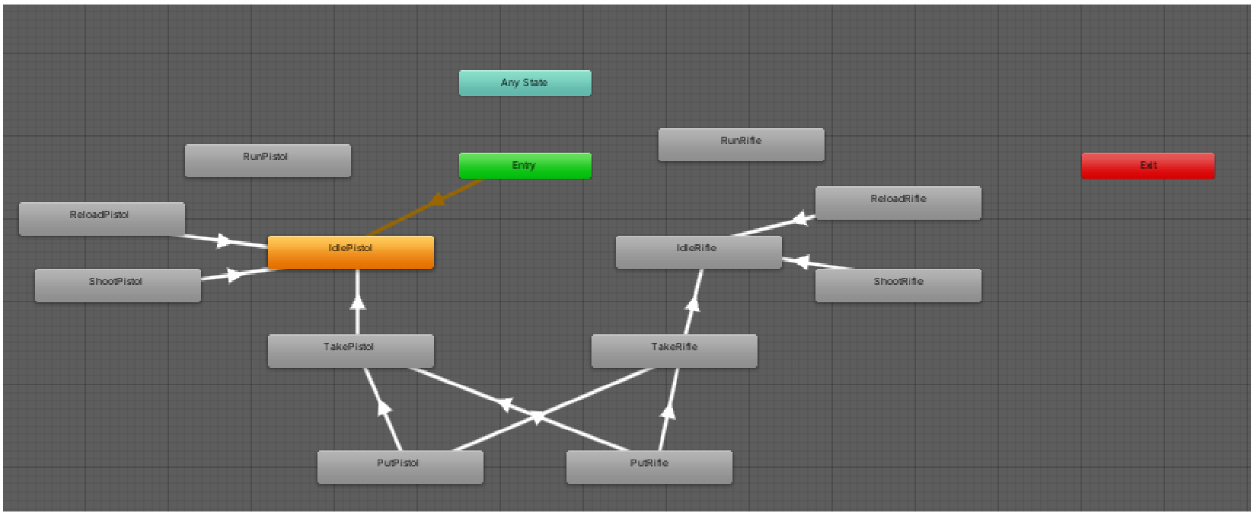


Figure 9. Player Animator

Task #3. *Create and/or animate 5 objects of your choice in your World (you can use Animator Component and/or Timeline)*

Book, which you can open up and close by being near and pressing “e”.

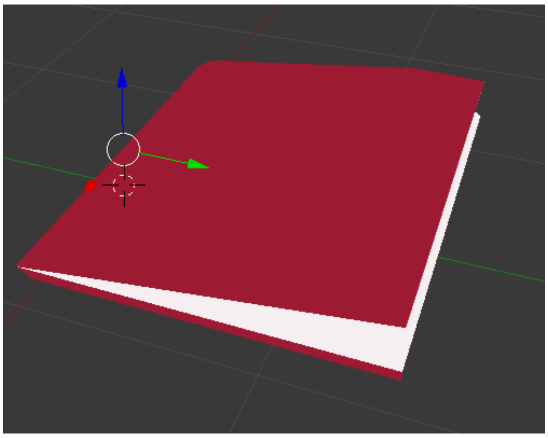


Figure 10. Animated Book.

Fireplace logs, animation can be activated by being nearby with “e”. This animation can only be activated one per session, it doesn’t reset.

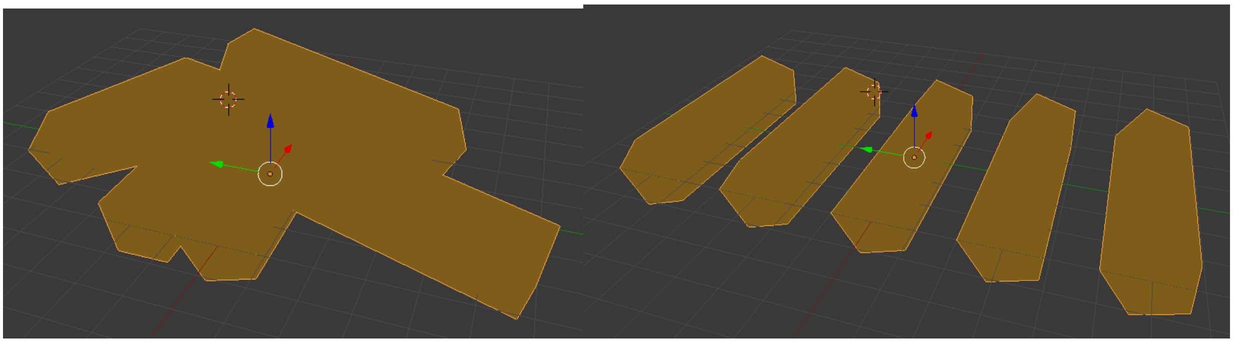


Figure 11. Logs in default form (rightside) and ready for fire (leftside).

Health box, this object is always playing spinning animation whit loop.

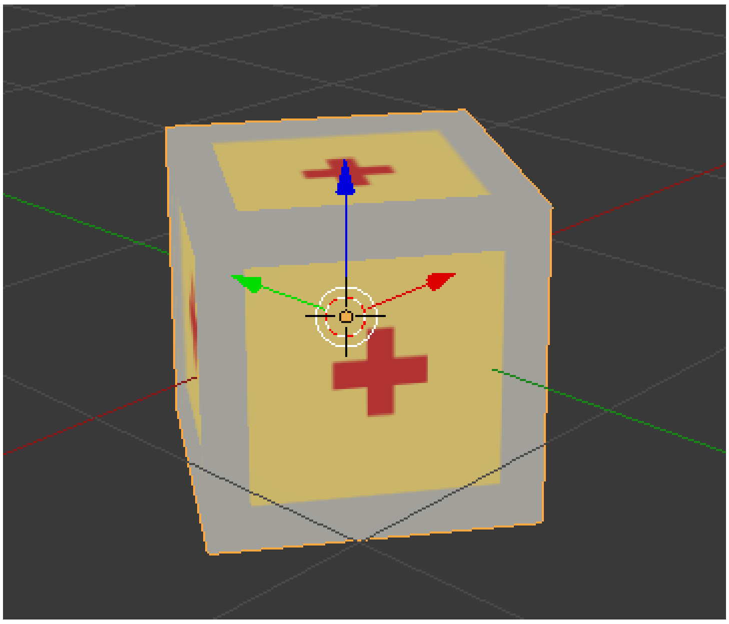


Figure 12. Health box.

Drinking duck, animation of this object is also activated by pressing “e”.

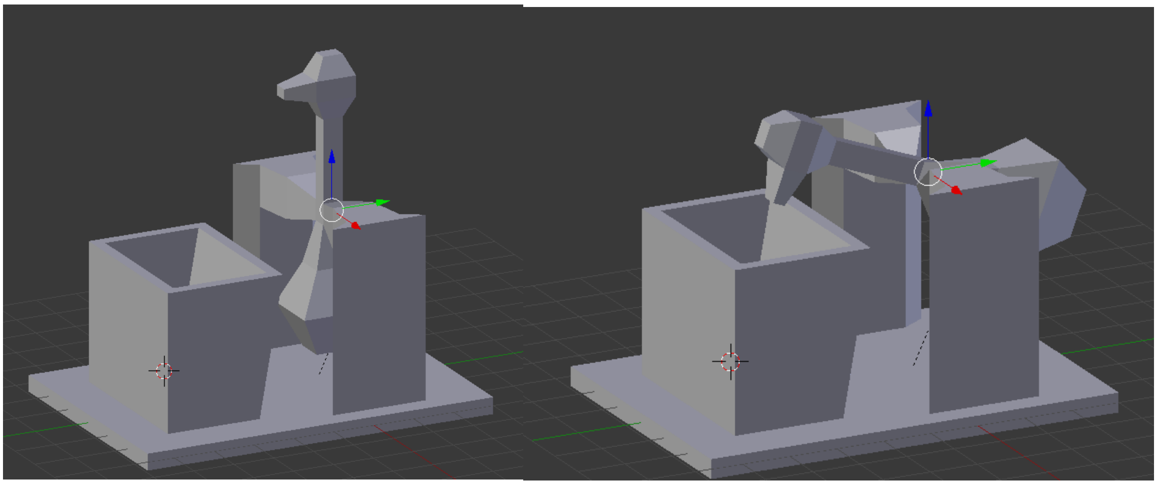


Figure 13. Drinking duck.

Vase, this object falls over, when touched. It can only be activated only once per session.

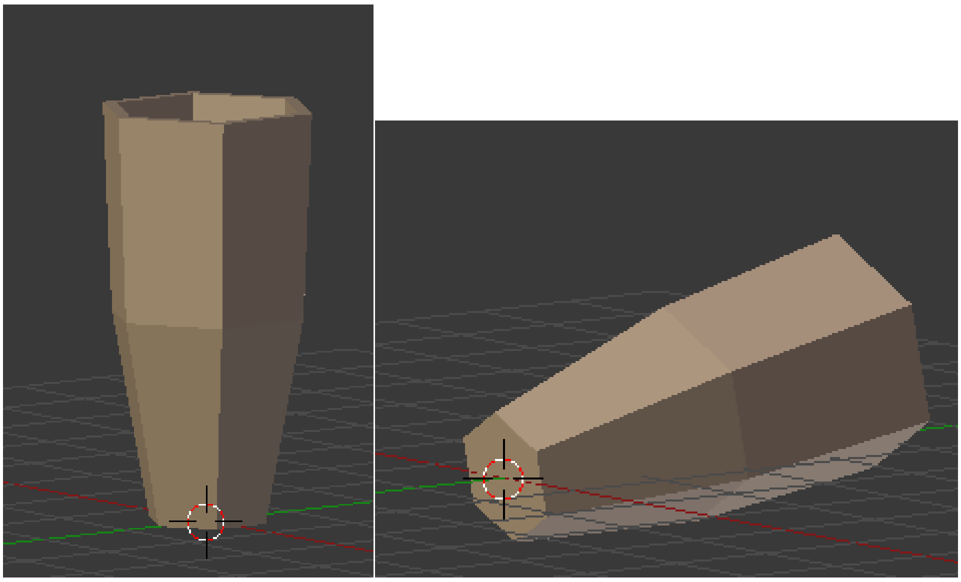


Figure 14. Vase with animation.

Task #4. *Create at least 5 particle effects for your environment (dust, explosion, smoke gas, light sparks, etc.)*

Light sparks, they often fall from the broken lamp.

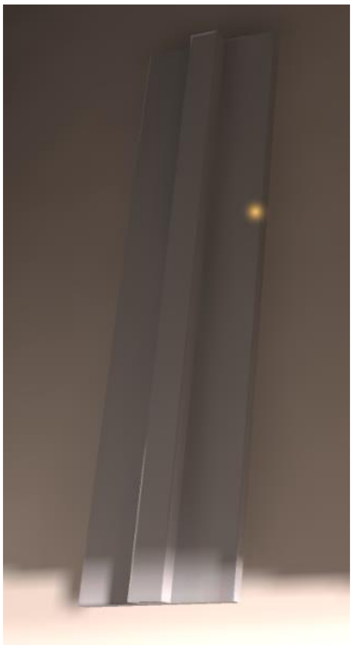


Figure 15. Light spark.

Smoke, it‘s hard to see it, but it‘s there.

Figure 16. Some smoke.

Fire particles, they appear when player activates fire place with “e”.

Figure 17. Fire particles.

Red smoke particle, it appears, when health box is picked up.

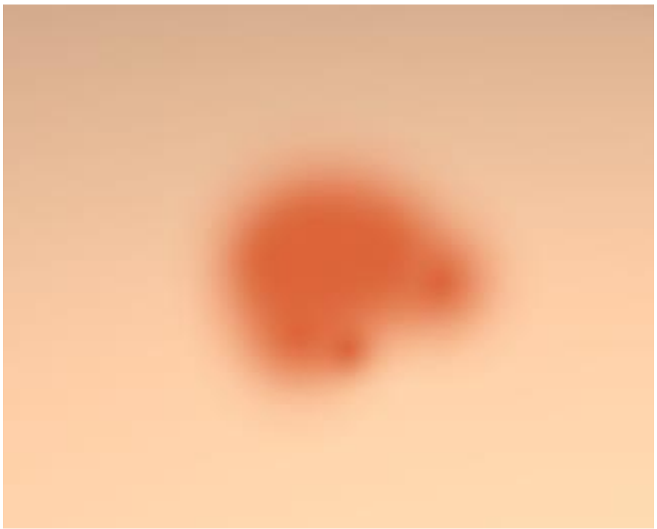
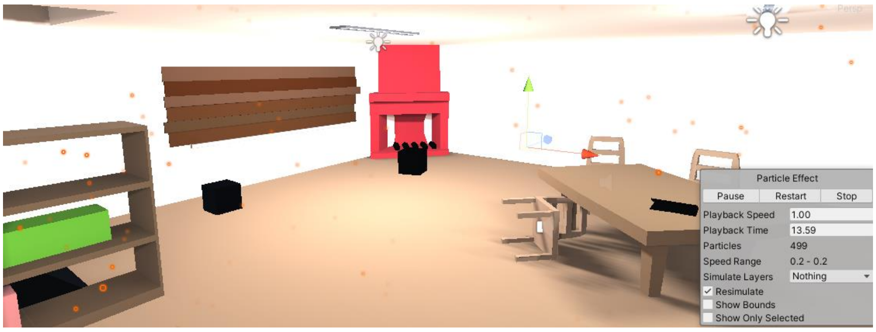


Figure 18. Red smoke particles.

Dust particles, they are always active in living room, though it‘s hard to see them.

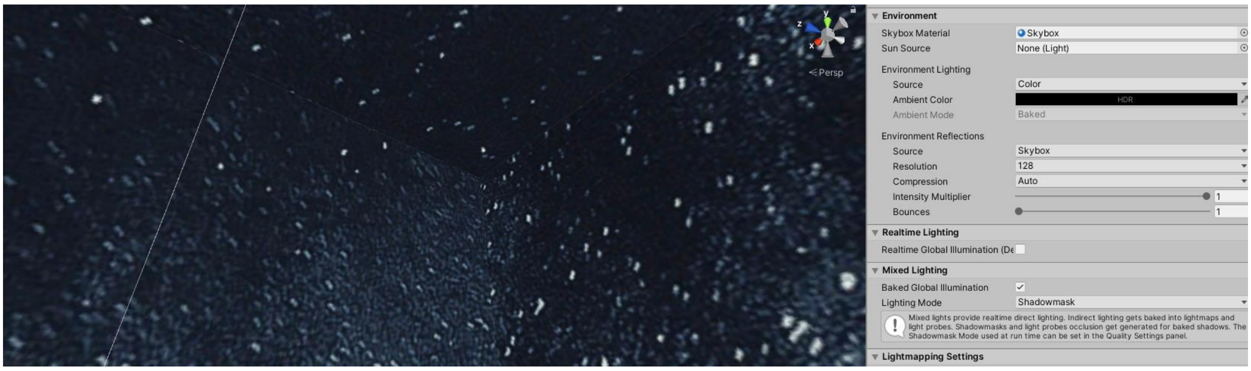


Playback Speed 1.00
Playback Time 13.59
Particles 499
Speed Range 0.2 - 0.2
Simulate Layers Nothing
☑ Resimulate
□ Show Bounds
□ Show Only Selected

Figure 19. Dust particles marked with outline.

Task #5. *Add a custom skybox*

Made a skybox, by creating a cube in blender, than adding texture to the cube and made 6 pictures inside that cube. Although skybox clearly isn't perfect, it won't affect game, because you have to play game inside a house.



▼ Environment

Skybox Material: Skybox

Sun Source: None (Light)

Environment Lighting

Source: Color

Ambient Color: HDR

Ambient Mode: Baked

Environment Reflections

Source: Skybox

Resolution: 128

Compression: Auto

Intensity Multiplier: 1

Bounces: 1

▼ Realtime Lighting

Realtime Global Illumination (De □

▼ Mixed Lighting

Baked Global Illumination ☑

Lighting Mode: Shadowmask

Mixed lights provide realtime direct lighting. Indirect lighting gets baked into lightmaps and light probes. Shadowmasks and light probes occlusion get generated for baked shadows. The Shadowmask Mode used at run time can be set in the Quality Settings panel.

▼ Lightmapping Settings

Figure 20. Custom skybox.

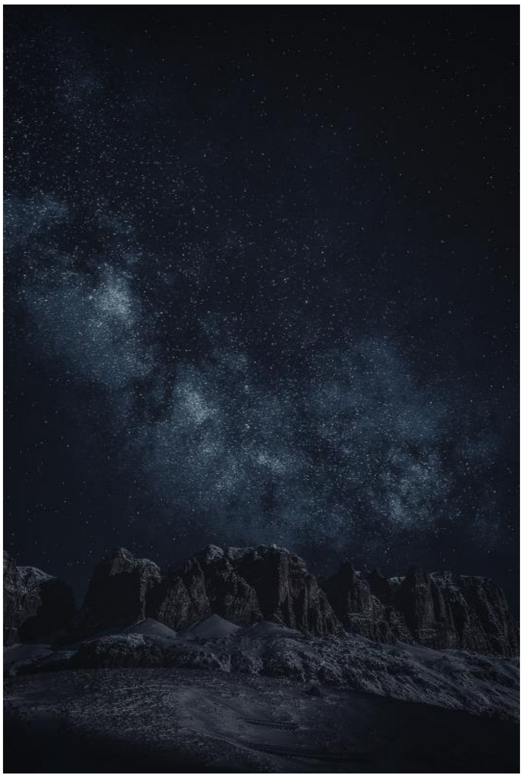


Figure 21. Picture used to make skybox.

Task #6. *Create at least 3 different Physics Materials for various parts of the map (either it's a slippery platform/ice, bouncy wall, non-slippery ground, etc.)*

Made a slippery platform on this object:

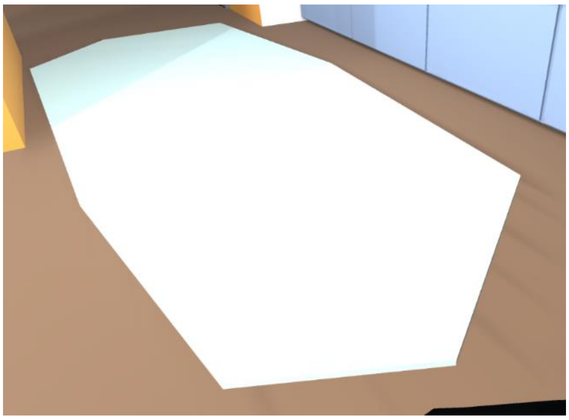
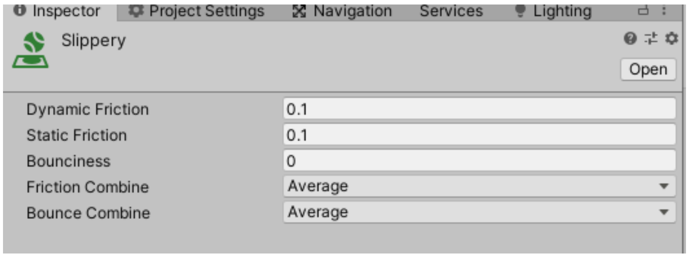


Figure 22. Slippery water.

Though it doesn‘t effect player a lot.

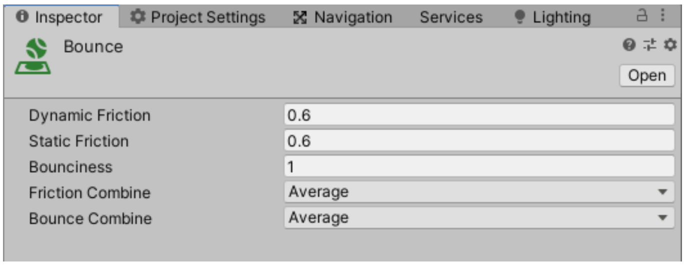


Slippery

Dynamic Friction	0.1
Static Friction	0.1
Bounciness	0
Friction Combine	Average
Bounce Combine	Average

Figure 23. Slippery physics material.

Made bouncy physics material, and added it to bed and a ball:



Bounce

Dynamic Friction	0.6
Static Friction	0.6
Bounciness	1
Friction Combine	Average
Bounce Combine	Average

Figure 24. Bouncy physics material.

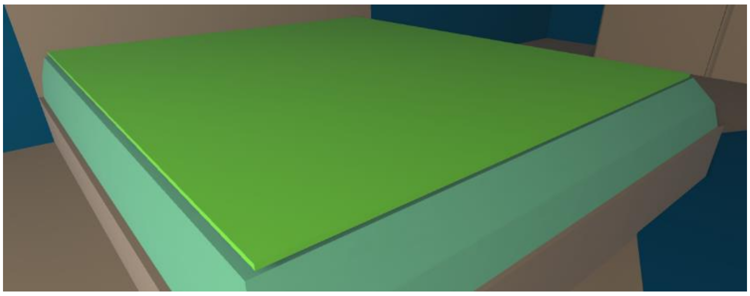
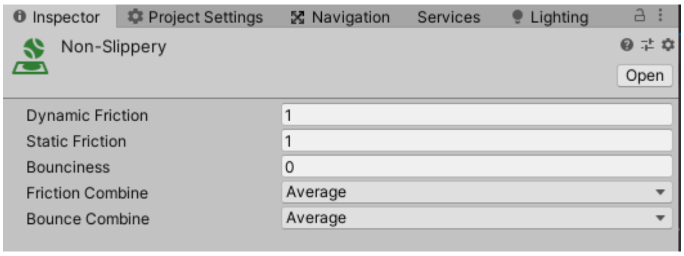


Figure 25. Green top has the bouncy physics material.

Figure 26. Ball that has bouncy physics material.

Non-slippery physics material – it doesn‘t really affect player.



Non-Slippery

Dynamic Friction 1
Static Friction 1
Bounciness 0
Friction Combine Average
Bounce Combine Average

Figure 27. Non-slippery physics material.

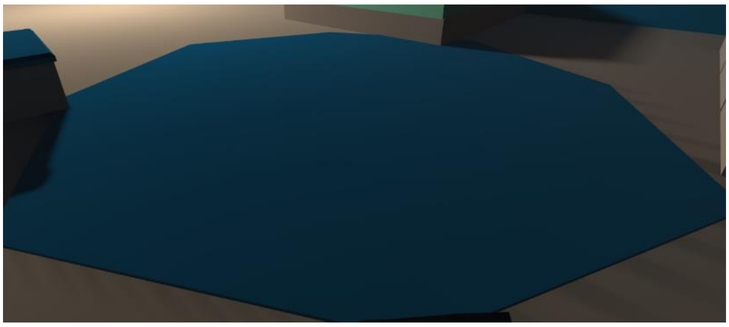


Figure 28. Object that has non-slippery material.

Task #7. *Create 4 types of objects that use OnCollisionEnter / OnCollisionEnter2D / OnTriggerEnter / OnTriggerEnter2D:*
–Destroys itself and spawns a particle system and adds a point;

Added healing box, which will later add health to bed, after touching this box it destroys itself and spawns red smoke particle.

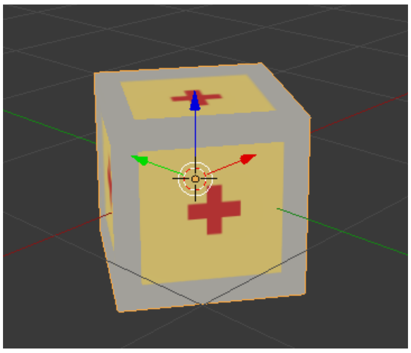


Figure 29. Box that has OnCollisionEnter fuction.

Figure 30. Red smoke particle that appears after destroying box.

–Plays a sound;

Chairs that play a sound after player collision:

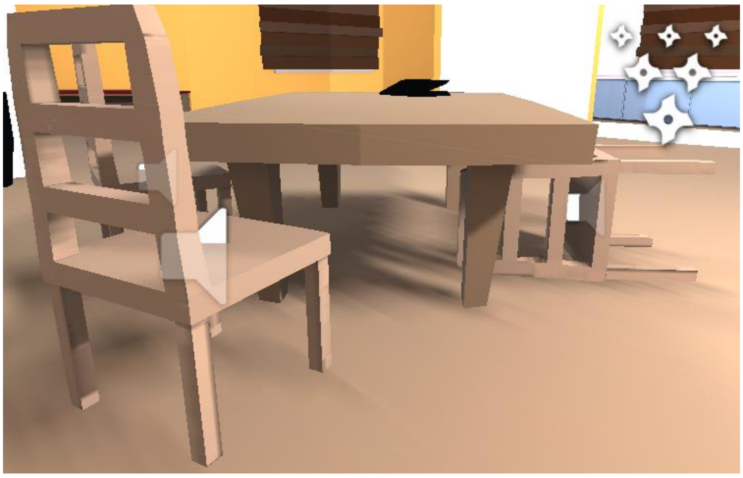


Figure 31. Chairs that have sound on collision.

–Moves away using Rigidbody Force or Velocity;

I added rigid body to a ball and after collision I added force to the rigid body.

Figure 32. Ball.

–Falls down / teleports / opens up (trap, door, etc.).

I made a vase, which plays animation when collided with and it falls over.

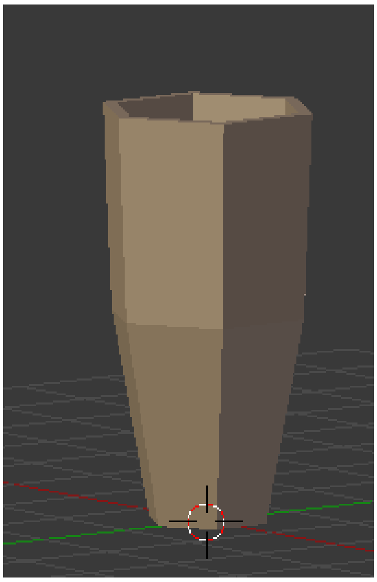


Figure 33. Vase with OnCollisionEnter function.

Task #8. *Assign optimal colliders for the environment objects that are not moving and set their flags*

For most of the objects, which were created in first laboratory work (Figure 2) I changed from mesh collider to box collider, because they mostly looked like boxes, some of them retained mesh colliders, but objects that can't be ducked under, hardly possible to jump on got box colliders. All of the objects that didn't have animation or some kind of movement got a static mark.

Before changing colliders average fps was 190, lowest was around 60 and peak about 260. After changing mesh colliders there was slight improvement: average 198, lowest 70 and peak 270. There wasn't much of a change, because scene isn't very big and objects are boxy.

Task #9. *Bake a lightmap and measure performance*

After baking light maps there was a bigger improvement, because game didn't need to calculate all shadows all the time. Average fps was 220, lowest 100 and peak was 275. So after baking a lightmap there was improvement.

Task #10. *Optimize all textures depending on their parameters and measure graphical memory load*

All of the imported textures already had generated mip maps, so I only compressed the textures to take up less space. There wasn't much of improvement, because the objects don't have a lot of polygons, so the textures aren't high quality.

Laboratory work #3

List of tasks

1. ***Add a MENU system to your game (New Game, Choose Level [If Applies], Options, About, Exit);***

2. Add OPTIONS (2 separate sound bars for music and effects);
3.Game must have a GUI (elements should vary based on your game: health bars, scores, money, resources, button to go back to main menu, etc.) ;
4. Add attack mechanics (might be replaced with puzzle mechanics, push off the road mechanics (racing game), etc.);
5. Implement Opponents (enemy moves towards you and tries to kill you, might be replaced with random puzzles, race opponents, etc.);
6. Add health / powerup mechanics and indication in the GUI;
7. Add scoring system (e.g., Easiest enemy – 100, most difficult – 500, powerup – 1000, total game time calculated, etc.);
8. Add "Game over" condition (once the game ends you should display a high score and reload/main menu buttons;
9. Add "Post Processing" (Blurring, blood screen, etc.);
10. Add Interactive sounds (Sound of step, shoot, die, scream, etc. - minimum 10 different) and music (1 for menu + minimum 3 for in-game).
11. Make a 5x5 chess board (white and black cubes).

Solution

Task #1. *Add a MENU system to your game (New Game, Choose Level [If Applies], Options, About, Exit);*

Made all of the mentioned buttons on different scenes.

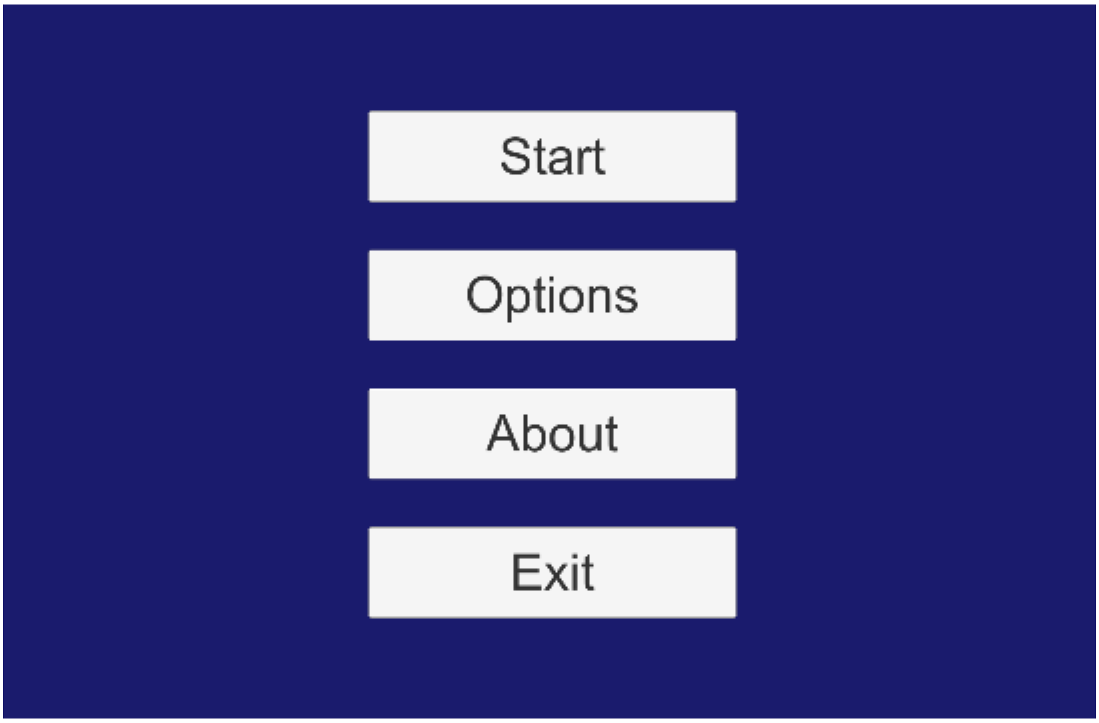


Figure 34. Main meniu buttons.

Start button goes trough scene manager and changes to main map. Options allows to change two parametrs. About button shows info who made the game and exit button leaves the game.

```
public void PlayGame()
{
SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
}

public void QuitGame()
{
Debug.Log("Quit");
Application.Quit();
}
```

Table 3. Code to change and to exit the game.

Task #2. *Add OPTIONS (2 separate sound bars for music and effects);*

As a task requires to change music and effects it was reached. Volume of sounds are manipulated with sliders.

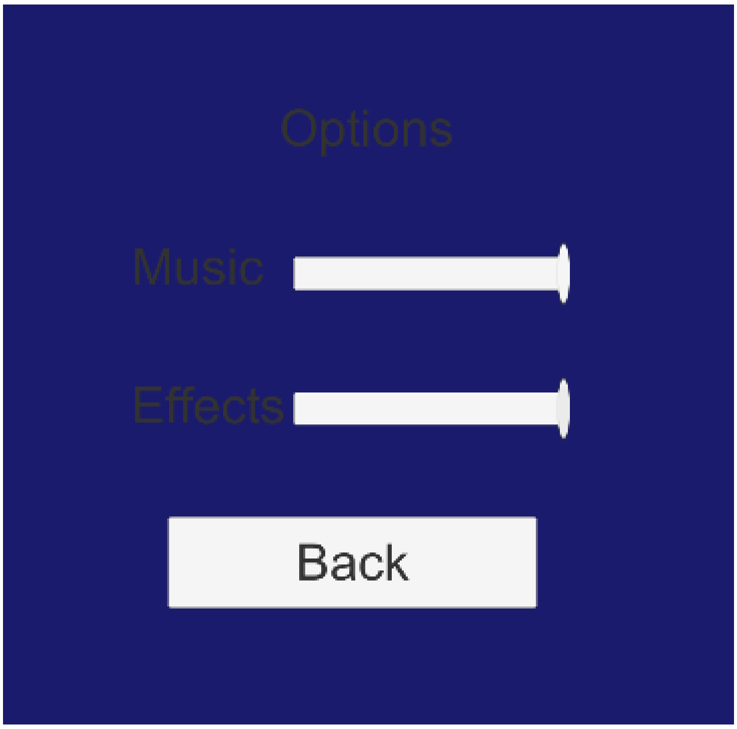


Figure 35 Options menu changing info with sliders.

Task #3. *Game must have a GUI (elements should vary based on your game: health bars, scores, money, resources, button to go back to main menu, etc.) ;*

On the players Screen game shows current players health (top left), ammo in current weapon (bellow players health), current score player has (bottom left), health of a bed (at the bottom center of view), money (top right) and info that you can start new wave with “Enter”.

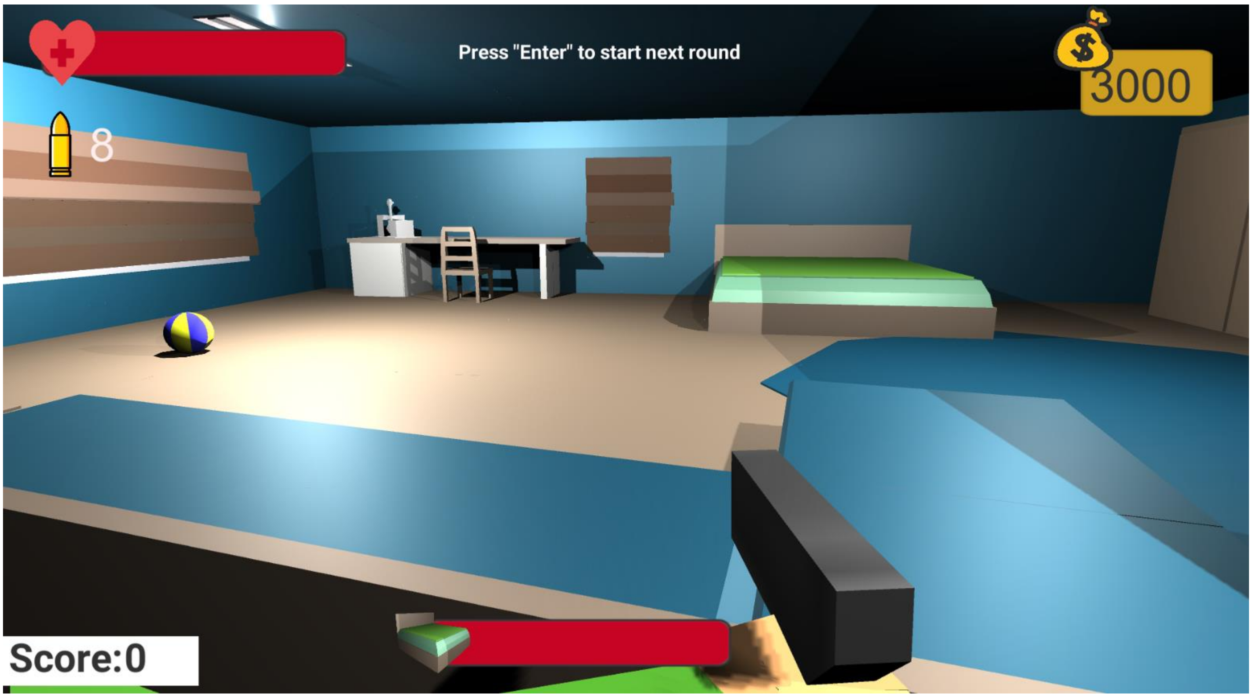
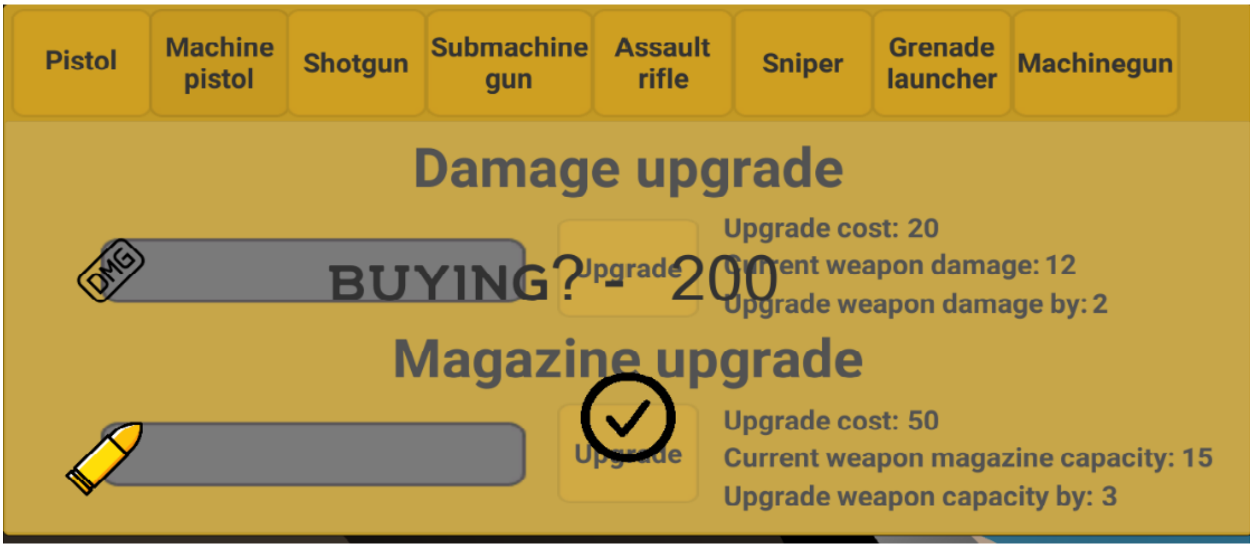


Figure 36. Players current interface.

Made a shop to buy and upgrade weapons with collected money. Each weapon can be upgraded 5 times.



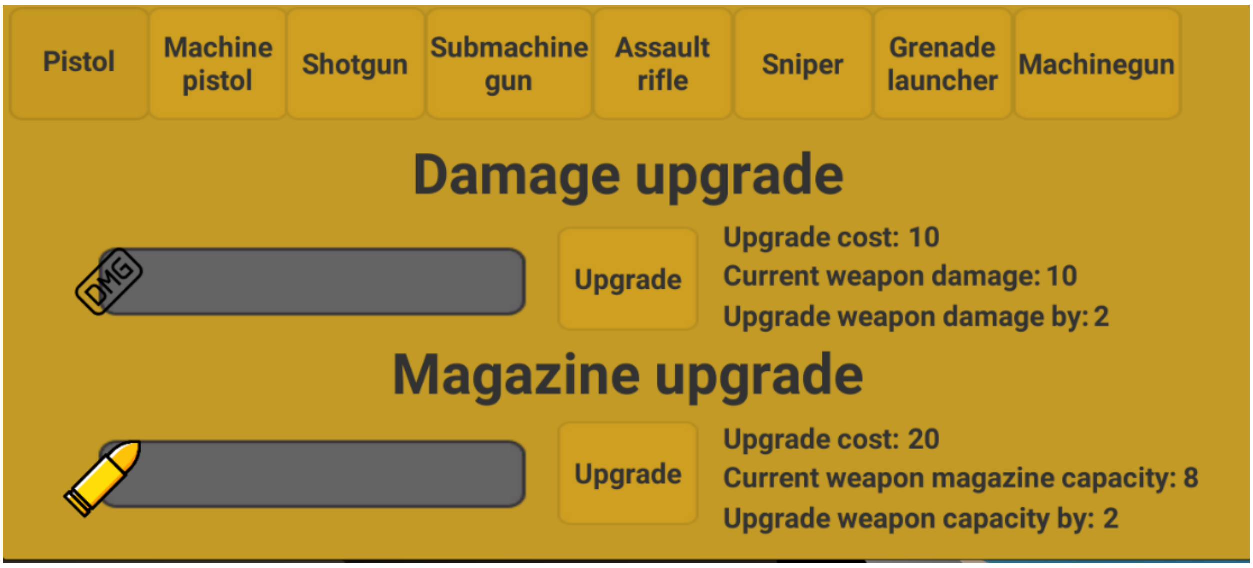
Damage upgrade

Upgrade cost: 20
Current weapon damage: 12
Upgrade weapon damage by: 2

Magazine upgrade

Upgrade cost: 50
Current weapon magazine capacity: 15
Upgrade weapon capacity by: 3

Figure 37. Buying new weapon by pressing check mark.



Damage upgrade

Upgrade cost: 10
Current weapon damage: 10
Upgrade weapon damage by: 2

Magazine upgrade

Upgrade cost: 20
Current weapon magazine capacity: 8
Upgrade weapon capacity by: 2

Figure 38. Weapon upgrade interface.

Game has pause menu too by setting time scale to zero almost all movement is stopped. Through pause menu player can go back to main menu, exit the game or resume.

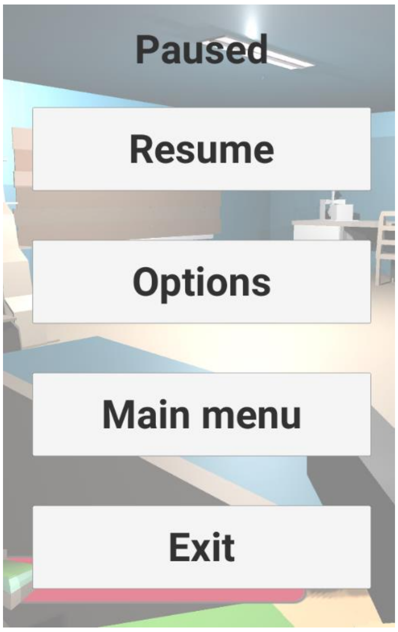


Figure 39. Pause menu.

Task #4. *Add attack mechanics (might be replaced with puzzle mechanics, push off the road mechanics (racing game), etc.);*

For attack mechanics I chose shooting weapons (pistol, rifles, etc) there are 8 in total weapons types. To shoot the weapon, weapon must have bullets in magazine, than player can shoot with mouse button. Program casts a raycast, which determines if player hit the enemy.

```
public void Shoot()
{
RaycastHit hit;
Physics.Raycast(MainCamera.transform.position, MainCamera.transform.forward, out hit);
if(hit.transform.tag == "Enemy")
if(currentWeapon != 6)
{
hit.transform.GetComponent<Enemy>().TakeDamage(weapons[currentWeapon].Damage);
}
else
{
Collider[] colliders;
colliders = Physics.OverlapSphere(hit.transform.position,
weapons[currentWeapon].ExplosionRadius);
foreach (Collider col in colliders)
if (col.tag == "Enemy")

col.GetComponent<Enemy>().TakeDamage(weapons[currentWeapon].Damage);
weapons[currentWeapon].CurrentAmmo--;
        bar.ChangeAmmoAmount(weapons[currentWeapon]
bar.ChangeAmmoAmount(weapons[currentWeapon].CurrentAmmo);
}
```

Table 4. Shoot() function.

Task #5. *Implement Opponents (enemy moves towards you and tries to kill you, might be replaced with random puzzles, race opponents, etc.);*

First target of zombies is a bed, but if player get's too close to a zombie, he will be targeted until player is out of detecting reach or dead. In code zombie will first calculate distance between zombie and bed, it will count distance between player and zombie aswell. Than zombie checks if either of the objects are close enough to perform attack, if there is no object within attacking distance that it will traverse toward a bed if player is not close enough.

```
distanceToPlayer = Vector3.Distance(player.transform.position,
transform.position);
distanceToBed = Vector3.Distance(bed.transform.position, transform.position);
if ((distanceToPlayer <= attackDistance || distanceToBed <= attackDistance + 1))
{
FaceTarget();
anim.Play("Attack" + Random.Range(1, 3))
}
else if (distanceToPlayer <= distance && lastTarget == bed)
MoveToTarget(player);
else if (Vector3.Distance(player.transform.position, transform.position) >=
distance && lastTarget == player)
MoveToTarget(bed);
```

Table 5. Zombie pathfinding

Task #6. *Add health / powerup mechanics and indication in the GUI;*

In this task I used object that was made for laboratory 1 and 2, just upgraded. After touching this object bed healing will be prioritized and only than player will helped.

```
if (core.FullHP())
bedHp.ChangeHp(2);
else
core.ChangeHp(2);
```

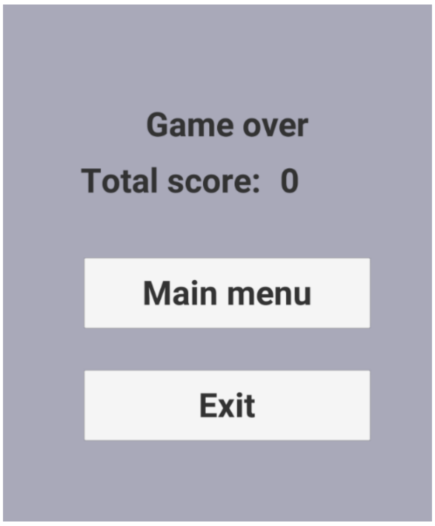
Table 6. Prioritizing bed health over princess.

Task #7. *Add scoring system (e.g., Easiest enemy – 100, most difficult – 500, powerup – 1000, total game time calculated, etc.);*

When a zombie is instantiated he will get random multiplier from 1 to 4, which determines if how much is the zombie worth. Base stat for points is 100, which is than multiplied with

Task #8. *Add "Game over" condition (once the game ends you should display a high score and reload/main menu buttons;*

When players or beds health reaches zero game is over, calculated info from task seven is displayed on game over screen. Than players only options is to go to main menu or exit the game.



Game over

Total score: 0

Figure 40. Game over screen.

Task #9. *Add "Post Processing" (Blurring, blood screen, etc.);*

Not implemented.

Task #10. *Add Interactive sounds (Sound of step, shoot, die, scream, etc. - minimum 10 different) and music (1 for menu + minimum 3 for in-game).*

All of the weapons got their own gun sound, when they shoot they play each own. There are two type of zombie textures, they both have own sound when they spawn. Main menu has own audio clip, which is always going. Main game scene has 3 types of tracks. First one is happier and is playing, when there are no enemies. Second track is a bit more aggressive and is playing, when there are zombies. Third track is being played, when player reaches wave 10.

Task #11. *Make a 5x5 chess board (white and black cubes).*

Made two cube prefabs. One with black material and one with white material. After that made a script with two “for” cycles to get 5x5 board. To know which prefab I needed made an “if” statement, which checks it by adding both “for” cycles current number and dividing them by two. Than checking if remainder equals to zero, than placing white prefab, if it was one, than placing black prefab.

```
void Start()
{
for(int i = 0; i < 5; i++)
for(int j = 0; j < 5; j++)
if((i + j) % 2 == 0)
Instantiate(whiteCube, new Vector3(i, 0, j), new Quaternion(0,0,0,0), gameObject.transform);
else Instantiate(blackCube, new Vector3(i, 0, j), new Quaternion(0, 0, 0, 0), gameObject.transform);
}
```

Table 7. Code to make 5x5 chess board

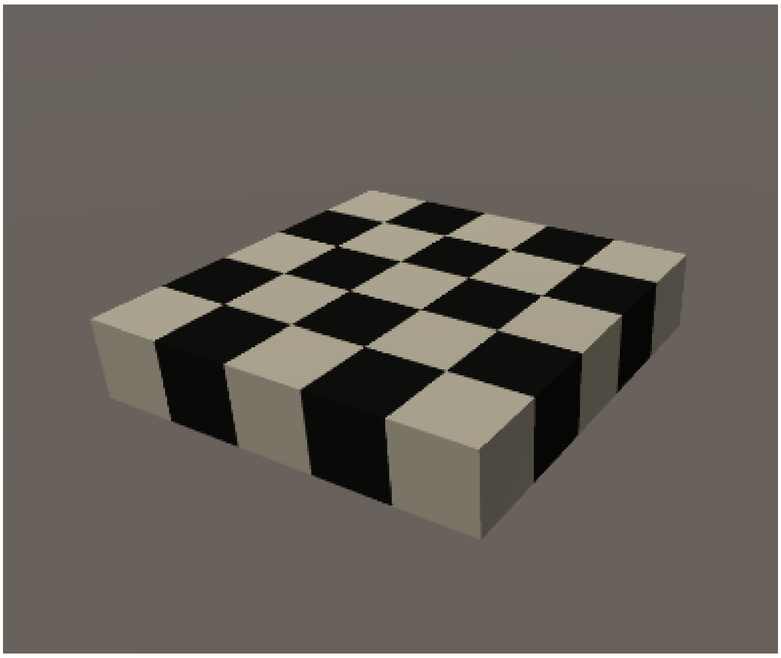


Figure 41. 5x5 chess board spawned with script.

User's manual (for the Individual work defence)

How to play? Idea of a game is for a Player to survive as long as possible. Player will have to not only defend himself from incoming zombie, but will have to defend a bed (Figure 41), which is on a second floor. Zombies only can enter your house from only two doors, which they will choose randomly.

Figure 42. Bed to defend.

Game will get harder as more waves passes on, there are going to be a lot of more zombies each wave. Player by killing zombies is going to get coins. With, which player will be able to buy or upgrade weapons. There total of eight weapons, some of them might deal more damage, but will hold less ammo. Addition to those eight weapons each of those weapons has five upgrades to damage and five upgrades to magazine capacity. So choose wisely.

Descriptions of the rules of the game.

- You have to survive as long as possible, to reach highest score.
- Attacking zombies first priority is your bed, but if you are close enough they will target player.

Descriptions of the controls / keys.

- W,S,A,D – walking around in the area;
- SpaceBar – jump;
- LShift – to move faster;
- Alpha keys one to eight are for changing current weapon (You won’t be able to change to a weapon you haven’t bought the weapon);
- Button “I” opens the shop inventory, there you can buy upgrades or weapons. Button “I” also closes the shop inventory.
- Escape button pauses the game and gives ability to return to main menu or exit the game itself.

Literature list

1. https://forum.unity.com/threads/how-do-i-clamp-a-quaternion.370041/
2. https://www.youtube.com/watch?v=tPtRCpwSgBg
3. https://www.youtube.com/watch?v=MYU1vBcUkfw
4. https://www.youtube.com/watch?v=DvKRGwCImJ4
5. https://www.youtube.com/watch?v=DnpoY8_T4H0
6. https://gamesounds.xyz/?dir=Sound%20Effects/Doors%20and%20Floors
7. https://assetstore.unity.com/packages/tools/gui/graphy-ultimate-fps-counter-stats-monitor-debugger-105778
8. https://opengameart.org/content/low-poly-soldier-with-weapons/
9. https://freesound.org/people/bolkmar/sounds/455922/
10. https://freesound.org/people/StephenSaldanha/sounds/132551/
11. https://freesound.org/people/nioczkus/sounds/378185/
12. https://freesound.org/people/LeMudCrab/sounds/163455/
13. https://freesound.org/people/GreenFireSound/sounds/396272/
14. https://freesound.org/people/SuperPhat/sounds/404562/
15. https://freesound.org/people/knova/sounds/172334/
16. https://freesound.org/people/tcpp/sounds/77340/
17. https://freesound.org/people/Under7dude/sounds/163439/
18. https://freesound.org/people/Breviceps/sounds/445983/
19. https://freesound.org/people/ShadyDave/sounds/345838/
20. https://freesound.org/people/ShadyDave/sounds/325611/
21. https://freesound.org/people/noirenex/sounds/159401/
22. https://freesound.org/people/noirenex/sounds/159398/
23. https://www.pngflow.com/ed9a78d5-8564-4055-a829-b1e8ad6a4c27
24. http://cdn.onlinewebfonts.com/svg/download_264720.png
25. https://www.flaticon.com/download-icon
26. https://freeicons.io/icon/download?_token=TTJF4jTSDcdzf7OVal4iOjeWPFuw4r9pOZqUySJB&icon_id=6941&type=original&size=128