Kode Forklaring

Indholdsfortegnelse

[Indexerings exemple 1](#_Toc505600049)

[1 Scripts 1](#_Toc505600050)

[1.1 Karakter og fjender 1](#_Toc505600051)

[1.1.1 Move 1](#_Toc505600052)

[1.1.2 Health 2](#_Toc505600053)

[1.1.3 Hitbox 2](#_Toc505600054)

[1.1.4 MovementAI 2](#_Toc505600055)

[1.1.5 TextScript 3](#_Toc505600056)

[1.2 Udstyr 3](#_Toc505600057)

[1.2.1 Bullet 3](#_Toc505600058)

[1.2.2 Weapon 3](#_Toc505600059)

[1.3 Ekstra 4](#_Toc505600060)

[1.3.1 ChildTriggerCollider 4](#_Toc505600061)

[2 Plugins 4](#_Toc505600062)

[2.1 Decals 4](#_Toc505600063)

[2.1.1 DecalObject 4](#_Toc505600064)

[2.1.2 BlitTexture2DExt 5](#_Toc505600065)

[2.1.3 DecalObjectEditor 5](#_Toc505600066)

[2.1.4 DecalWindow 5](#_Toc505600067)

[2.2 PrefabChanger 6](#_Toc505600068)

[2.2.1 PrefabChangerWindow 6](#_Toc505600069)

[2.2.2 PrefabOption 6](#_Toc505600070)

[2.2.3 ButtonInfo 6](#_Toc505600071)

[2.3 PrefabPlacer 7](#_Toc505600072)

[2.3.1 PrefabPlacerWindow 7](#_Toc505600073)

[2.4 MaterialReplacer 7](#_Toc505600074)

[2.4.1 MaterialReplacerWindow 7](#_Toc505600075)

[2.4.1.3.3 SetReplaceMaterial() 8](#_Toc505600076)

[2.5 RandomHouseColor 8](#_Toc505600077)

[2.5.1 RandomHouseColorParent 8](#_Toc505600078)

[2.5.2 RandomHouseColorEditor 8](#_Toc505600079)

# Indexerings exemple

Vores indexering at x er et tal. Bid mærke i at unity, public og private allrede har et bestemt tal.

**X Hoved Kategori**

**X.X Under Kategori**

**X.X.X Klasse Navn**

X.X.X.1.X Unity Events

X.X.X.2.X Public Method

X.X.X.3.X Private Method

# 1 Scripts

## 1.1 Karakter og fjender

### 1.1.1 Move

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Move : MonoBehaviour

#### 1.1.1.0.1Attributter

public float speed = 2.0f;

public float sensitivity = 2.0f;

public float gravity = 20.0f;

public float jumpSpeed = 8.0f;

public float maxHeight = 1.6f;

public float minHeight = 1.3f;

public float heightSmooth = 5f;

public float smoothing = 2.0f;

private Vector2 mouseLock;

private Vector2 smoothV;

private CharacterController player;

private Camera eyes;

private Vector3 movement = Vector3.zero;

private float crouchR = 1f;

private float moveFB;

private float moveLR;

private float rotX;

private float rotY;

private bool crouch;

#### 1.1.1.1.1 Start()

void Start()

{

player = GetComponent<CharacterController>();

eyes = Camera.main;

IsMouseLocked(true);

}

#### 1.1.1.1.2Update()

void Update()

{

RotateView();

CalculateMovement();

if (player.isGrounded && Input.GetButton("Jump"))

{

movement.y = jumpSpeed;

}

if (Input.GetButton("Crouch") || !CanStand())

{

StartCrouching();

}

else

{

StopCrouching();

}

if (Input.GetKeyUp(KeyCode.I))

{

IsMouseLocked(Cursor.visible);

}

if (Input.GetKeyUp(KeyCode.P))

{

Time.timeScale = 0.1f;

}

if (Input.GetKeyUp(KeyCode.O))

{

Time.timeScale = 1;

}

//Debug.DrawLine(eyes.transform.position, eyes.transform.forward \* 20f, Color.red);

movement.y -= gravity \* Time.deltaTime;

player.Move(movement \* Time.deltaTime);

}

#### 1.1.1.3.1 CalculateMovement()

private void CalculateMovement()

{

moveFB = Input.GetAxis("Vertical") \* speed \* crouchR;

moveLR = Input.GetAxis("Horizontal") \* speed \* crouchR;

movement = new Vector3(moveLR, movement.y, moveFB);

// Gør at tasterne passer iforhold til hvor vi kigger

movement = transform.rotation \* movement;

}

#### 1.1.1.3.2 RotateView()

private void RotateView()

{

var md = new Vector2(Input.GetAxis("Mouse X"), Input.GetAxis("Mouse Y"));

// Ganger den første vector2 x og y med den anden vector2 x og y

md = Vector2.Scale(md, new Vector2(sensitivity \* smoothing, sensitivity \* smoothing));

// Gør at man hurtigt langsomt stopper hovedet (som hvis du fader noget ud)

smoothV.x = Mathf.Lerp(smoothV.x, md.x, 1f / smoothing);

smoothV.y = Mathf.Lerp(smoothV.y, md.y, 1f / smoothing);

// Gør at vi ikke kan lave en 360 rundt nede og oppe

mouseLock += smoothV;

mouseLock.y = Mathf.Clamp(mouseLock.y, -75f, 75f);

// Her rotere vi kameraet og kroppe

eyes.transform.localRotation = Quaternion.AngleAxis(-mouseLock.y, Vector3.right);

player.transform.localRotation = Quaternion.AngleAxis(mouseLock.x, player.transform.up);

}

#### 1.1.1.3.3 StartCrouching()

private void StartCrouching()

{

var value = heightSmooth \* Time.deltaTime;

var center = player.center;

crouchR = 0.6f;

// her sørger vi får at det tager tid for at dykke sig

if (player.height > minHeight)

{

player.height -= value;

center.y += value / 2;

}

// Her sætter vi character controllerns mide og højte så den passer til character

if (player.height < minHeight)

{

center.y -= (minHeight - player.height) / 2;

player.height = minHeight;

}

player.center = center;

}

#### 1.1.1.3.4 StopCrouching()

private void StopCrouching()

{

var value = heightSmooth \* Time.deltaTime;

var center = player.center;

crouchR = 1f;

// her sørger vi får at det tager tid for at rejse sig

if (player.height < maxHeight)

{

player.height += value;

center.y -= value / 2;

}

// Her sætter vi character controllerns mide og højte så den passer til character

if (player.height > maxHeight)

{

center.y += (player.height - maxHeight) / 2;

player.height = maxHeight;

}

player.center = center;

}

#### 1.1.1.3.5 CanStand()

private bool CanStand()

{

float move = player.radius - 0.1f;

for (int i = 0; i < 4; i++)

{

var posTop = transform.position;

posTop.y += maxHeight - 0.1f;

// Tjekker max 4 raycast om der er noget over dem, hvis det er noget over dem kan vi ikke rejse os

switch (i)

{

case 0: posTop.x += move; break;

case 1: posTop.x -= move; break;

case 2: posTop.z += move; break;

case 3: posTop.z -= move; break;

}

// Her tjekker vi på om nogen af dem er false

if (Physics.Raycast(posTop, Vector3.up, maxHeight - player.height))

return false;

}

return true;

}

#### 1.1.1.3.6 IsMouseLocked()

private void IsMouseLocked(bool val)

{

Cursor.visible = !val;

Cursor.lockState = val ? CursorLockMode.Locked : CursorLockMode.None;

}

### 1.1.2 Health

using UnityEngine;

using UnityEngine.UI;

public class Health : MonoBehaviour

#### 1.1.2.0.1 Attributter

public bool IsAlive { get { return currentHP > 0; } }

public bool UseHealthBar { get { return healthBarTemp != null; } }

public Canvas healthBarTemp;

public Vector3 healthBarPosition;

public GameObject[] hitboxs;

public int currentHP;

public int maxHP = 500;

private Canvas healthBar;

private Slider healthBarSlider;

#### 1.1.2.1.1 Start()

void Start()

{

currentHP = maxHP;

for (int i = 0; i < hitboxs.Length; i++)

hitboxs[i].GetComponent<Hitbox>().OnTakeDamge = TakeDamage;

if (UseHealthBar)

{

healthBar = Instantiate<Canvas>(healthBarTemp);

healthBar.transform.SetParent(transform);

healthBar.transform.localPosition = healthBarPosition;

healthBarSlider = healthBar.GetComponentInChildren<Slider>();

healthBarSlider.maxValue = maxHP;

healthBarSlider.value = currentHP;

}

}

#### 1.1.2.1.2 Update()

void Update() { if (!IsAlive && gameObject.activeSelf) gameObject.SetActive(false); }

#### 1.1.2.2.1 TakeDamage()

public void TakeDamage(int dmg)

{

currentHP -= dmg < currentHP ? dmg : currentHP;

if (UseHealthBar)

{

healthBarSlider.value = currentHP;

}

}

### 1.1.3 Hitbox

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[RequireComponent(typeof(Collider))]

public class Hitbox : MonoBehaviour {

#### 1.1.3.0.1 Attributter

public delegate void TakeDamge(int dmg);

public TakeDamge OnTakeDamge { private get; set; }

public float percent = 1;

private LayerMask CanGiveDamge;

#### 1.1.3.1.1 Start()

void Start ()

{

CanGiveDamge = 1 << LayerMask.NameToLayer("Bullet") | 1 << LayerMask.NameToLayer("Main Player");

}

#### 1.1.3.2.1 OnGameObjectEnter()

void OnGameObjectEnter(GameObject gb)

{

//Debug.Log("Hit: " + gb.layer);

if ((CanGiveDamge.value & (1 << gb.layer)) == (1 << gb.layer))

{

//Debug.Log("Hit: " + gb.activeSelf);

var bullet = gb.GetComponent<Bullet>();

if (!bullet.hasHitTarget || true)

{

bullet.hasHitTarget = true;

OnTakeDamge(Mathf.RoundToInt(bullet.dmg \* percent));

}

}

}

### 1.1.4 MovementAI

using System.Collections;

using System.Collections.Generic;

using UnityEngine.AI;

using UnityEngine;

[RequireComponent(typeof(NavMeshAgent))]

public class MovementAI : MonoBehaviour

#### 1.1.4.0.1 Attributter

public LayerMask followTarget;

[Range(0f, 360f)]

public float fieldOfViewDegrees;

public float radius;

private Vector3 startPoint;

private Vector3 endPoint;

private Vector3 lastPoint;

private SphereCollider sphereCollider;

private NavMeshAgent enemy;

private Animator anim;

private bool isFollowTarget;

#### 1.1.4.1.1 Start()

void Start()

{

anim = GetComponent<Animator>();

enemy = gameObject.GetComponent<NavMeshAgent>();

startPoint = transform.position;

var targetRange = new GameObject();

targetRange.name = "TriggerRange";

targetRange.layer = LayerMask.NameToLayer("Trigger");

targetRange.transform.parent = transform;

targetRange.transform.position = transform.position;

var sphereCollider = targetRange.AddComponent<SphereCollider>();

sphereCollider.isTrigger = true;

sphereCollider.radius = radius;

var targetTrigger = targetRange.AddComponent<ChildTriggerCollider>();

targetTrigger.TriggerOnStay = TriggerOnStay;

}

#### 1.1.4.1.2 Update()

void Update()

{

if (Input.GetKeyDown("h"))

{

anim.SetInteger("Test1", 1);

}

else if (Input.GetKeyDown("j"))

{

anim.SetInteger("Test1", 2);

}

if (Input.GetKeyDown("k"))

{

anim.SetInteger("Test1", 3);

}

if (Input.GetKeyDown("l"))

{

anim.SetInteger("Test1", 4);

}

//if (lastPoint == transform.position || enemy.isStopped || endPoint == null

// || layerMask.value == 1 << LayerMask.NameToLayer("Main Player"))

//{

// anim.SetInteger("Test1", 3);

//}

if (lastPoint == transform.position || enemy.isStopped || endPoint == null || Vector3.Distance(transform.position, endPoint) < 10)

{

RandomMove();

anim.SetInteger("Test1", 1);

}

lastPoint = transform.position;

//RaycastHit hit;

//var players = Physics.OverlapSphere(transform.position, radius, layerMask);

//if (players.Length > 0)

//{

// Debug.Log(players[0].transform.position);

// enemy.SetDestination(players[0].transform.position);

//}

}

#### 1.1.4.3.1 TriggerOnStay()

private void TriggerOnStay(Collider col)

{

// Tjekker om layer mask er et match og om noget er foran den

if ((followTarget.value & 1 << col.gameObject.layer) == (1 << col.gameObject.layer) && CanSeePlayer(col.gameObject))

enemy.SetDestination(col.transform.position);

}

#### 1.1.4.3.2 RandomMove()

private void RandomMove()

{

NavMeshHit hit;

Vector3 rDirection = Random.insideUnitSphere \* radius;

rDirection += startPoint;

NavMesh.SamplePosition(rDirection, out hit, radius, 1);

enemy.SetDestination(hit.position);

endPoint = hit.position;

}

#### 1.1.4.3.3 CanSeePlayer()

private bool CanSeePlayer(GameObject gb)

{

RaycastHit hit;

Vector3 rayDirection = gb.transform.position - transform.position;

// Tjekker på om der er noget inden for den angle som vi laver

if ((Vector3.Angle(rayDirection, transform.forward)) <= fieldOfViewDegrees \* 0.5f)

if (Physics.Raycast(transform.position, rayDirection, out hit, radius))

return (hit.transform.CompareTag("Player"));

return false;

}

### 1.1.5 TextScript

using System.Collections;

using System.Collections.Generic;

using UnityEngine.UI;

using UnityEngine;

public class TextScript : MonoBehaviour {

#### 1.1.5.0.1 Attributter

public Text health\_text;

public Text ammo\_text;

private Health health;

private Weapon ammo;

#### 1.1.5.1.1 Start()

void Start ()

{

health = GetComponentInParent<Health>();

ammo = GetComponentInParent<Weapon>();

}

#### 1.1.5.1.2 Update()

void Update ()

{

return;

string allHealth = health.currentHP.ToString() + "/" + health.maxHP.ToString();

health\_text.text = allHealth;

string allAmmo = ammo.CurrentShots.ToString() + "/" + ammo.CurrentAmmo.ToString();

ammo\_text.text = allAmmo;

}

## 1.2 Udstyr

### 1.2.1 Bullet

using UnityEngine;

[RequireComponent(typeof(Rigidbody))]

[RequireComponent(typeof(Collider))]

public class Bullet : MonoBehaviour

#### 1.2.1.0.1 Attributter

public Camera PlayerEyes { private get; set; }

public LayerMask ignoreCollision;

public bool hasHitTarget;

public float range = 50;

public float speed = 70f;

public float dmg = 50;

private Vector3 startPoint;

private Vector3 endPoint;

private LayerMask ignoreMask;

private Vector3? lastPosition;

#### 1.2.1.1.1 Update()

void Update ()

{

// Så længe gameObject ikke er synlig skal vi ikke gøre noget

if (!gameObject.activeSelf) return;

// Da vores gameobject kan "teleport"(hvis den har meget speed) bliver vi nød til

// at tjekke om vi ville have ramt noget i mellem den gamle

// position til den nye position

RaycastHit hit;

if (lastPosition.HasValue && Physics.Linecast(lastPosition.Value, transform.position, out hit, ~ignoreCollision.value))

{

// Gør gameObject usynlig

gameObject.SetActive(false);

// Hvis hit har Metoden OnGameObjectEnter() på sig i et

// script sender vi dette gameObject som parameter

hit.collider.SendMessage("OnGameObjectEnter", gameObject, SendMessageOptions.DontRequireReceiver);

}

// Flyt skudet/Sæt ny position

Vector3 heading = endPoint - startPoint;

Vector3 direction = heading / heading.magnitude;

float distanceThisFrame = speed \* Time.deltaTime;

//transform.Translate(direction.normalized \* distanceThisFrame, Space.World);

transform.position += direction \* speed \* Time.deltaTime;

// Gør gameObject usynlig hvis den er ude af range

if (Vector3.Distance(startPoint, transform.position) > range)

gameObject.SetActive(false);

// Opdater position

lastPosition = transform.position;

}

#### 1.2.1.1.2 OnTriggerEnter()

void OnTriggerEnter(Collider collision)

{

// Vores gameobject skal være synlig så vi ikke kan kalde den dobbelt. (Kan kaldes fra Update())

if (gameObject.activeSelf && (ignoreCollision.value & (1 << collision.gameObject.layer)) != (1 << collision.gameObject.layer))

{

collision.gameObject.SendMessage("OnGameObjectEnter", gameObject, SendMessageOptions.DontRequireReceiver);

gameObject.SetActive(false);

}

}

#### 1.2.1.1.3 OnEnable()

void OnEnable()

{

//startPoint = transform.position;

lastPosition = startPoint;

hasHitTarget = false;

}

#### 1.2.1.3.1 Fire()

public void Fire(Vector3 startPos)

{

//Sæt start position

startPoint = startPos;

transform.position = startPoint;

// Henter kameraes position ud fra "verden" og

// Vector3 variablen gør at den tager fra midten af skærmen

Vector3 rayOrigin = PlayerEyes.ViewportToWorldPoint(new Vector3(0.5f, 0.5f, 0));

// Tjekker om der er et object foran vores skyd, hvis ja får vi af vide hvad, hvis nej flyver skydet bare lige ud

RaycastHit hit;

bool hitTarget = Physics.Raycast(rayOrigin, PlayerEyes.transform.forward, out hit, range, ~ignoreCollision);

endPoint = hitTarget ? hit.point : rayOrigin + PlayerEyes.transform.forward \* range;

if (hitTarget) { Debug.Log("Hit T: " + hit.collider.name); }

//Gør synlig

gameObject.SetActive(true);

}

### 1.2.2 Weapon

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Weapon : MonoBehaviour

#### 1.2.2.0.1 Attributter

public int CurrentAmmo { get; private set; }

public int CurrentShots { get; private set; }

private bool HasShots { get { return CurrentShots > 0; } }

private bool CanReload { get { return CurrentShots != shots && CurrentAmmo > 0; } }

public Transform gunEnd;

public GameObject bulletTemplate;

public float weaponRange = 10f;

public int shots = 30;

public int ammo = 1000;

public float reloadSpeed;

public float fireRate = 0.25f;

private List<GameObject> bullets = new List<GameObject>();

private LineRenderer laserLine;

private Camera eyes;

private WaitForSeconds shotDuration = new WaitForSeconds(0.07f);

private float nextFire;

private bool isReloading = false;

#### 1.2.2.1.1 Start()

void Start()

{

eyes = Camera.main;

CurrentShots = shots;

CurrentAmmo = ammo;

}

#### 1.2.2.1.2 Update()

void Update()

{

if (!isReloading && CanReload && (Input.GetButtonUp("Reload") || CurrentShots == 0))

{

// Starter Reload() så den kan blive paused når som helst

StartCoroutine(Reload());

}

}

#### 1.2.2.1.3 LateUpdate()

void LateUpdate()

{

if (Input.GetButton("Fire1") && !isReloading && Time.time > nextFire && HasShots)

{

// Laver en begrænsning for hvornår man kan skyde igen

nextFire = Time.time + fireRate;

GetBullet().GetComponent<Bullet>().Fire(gunEnd.position);

CurrentShots--;

}

}

#### 1.2.2.3.1 GetBullet()

private GameObject GetBullet()

{

// Her ser vi om der er et usynligt bullet, hvis ja sender vi det tilbage

foreach (var bullet in bullets)

if (!bullet.activeSelf)

return bullet;

var newBullet = Instantiate(bulletTemplate);

newBullet.GetComponent<Bullet>().PlayerEyes = eyes;

// Sørger for at collideren på vores bullet og våben ikke kan ramme ind i hinanden

Physics.IgnoreCollision(newBullet.GetComponent<Collider>(), gameObject.GetComponent<Collider>());

bullets.Add(newBullet);

return newBullet;

}

#### 1.2.2.3.2 Reload()

private IEnumerator Reload()

{

isReloading = true;

//Debug.Log("Reload ");

// laver en time på 2 sek

yield return new WaitForSeconds(2f);

int cAmmo = CurrentAmmo;

int cShots = CurrentShots;

// sørger for at vi får fuld reload hvis der er nok skyd til det eller giver os det som er tilbage

if (CurrentAmmo + CurrentShots - shots > 0)

{

CurrentAmmo = CurrentAmmo + CurrentShots - shots;

CurrentShots = shots;

}

else

{

CurrentShots = CurrentAmmo + CurrentShots;

CurrentAmmo = 0;

}

//Debug.Log("shot " + CurrentShots);

//Debug.Log("ammo " + CurrentAmmo);

isReloading = false;

}

## 1.3 Ekstra

### 1.3.1 ChildTriggerCollider

using UnityEngine;

public class ChildTriggerCollider : MonoBehaviour

#### 1.3.1.0.1 Attributter

public delegate void OnTrigger(Collider collider);

public OnTrigger TriggerOnEnter { private get; set; }

public OnTrigger TriggerOnStay { private get; set; }

public OnTrigger TriggerOnExit { private get; set; }

#### 1.3.1.1.1 OnTriggerEnter()

void OnTriggerEnter(Collider collider) { if (TriggerOnEnter != null) TriggerOnEnter(collider); }

#### 1.3.1.1.2 OnTriggerStay()

void OnTriggerStay(Collider collider) { if (TriggerOnStay != null) TriggerOnStay(collider); }

#### 1.3.1.1.3 OnTriggerExit()

void OnTriggerExit(Collider collider) { if (TriggerOnExit != null) TriggerOnExit(collider); }

# 2 Plugins

## 2.1 Decals

### 2.1.1 DecalObject

Bruges ikke mere da vi fandt ud af at have for mange decals/gameobject fik spillet til at lagge vi har løst problemet ved at lave en shader i stedet for.

Dette script gør kun noget i editor mode og det den gør er at den laver en Quad hvor der bliver sat et sprite på. Dette script bruger også en CustomEditor (DecalObjectEditor.cs)

#### 2.1.1.1.1 OnEnable()

Opdatere decal når man starter Unity editor. Der er en fejl der gør at den også bliver kaldt Når man bare testet spillet

#### 2.1.1.2.1 BuildDecal()

Her laver vi gameobjectet om til en decal med billede. Der bliver ikke brugt shared material (kan være en grund til at det lagger) hvis den er tæt på et andet gameobject vil den tegne billedet på det andet gameobject (Der er et problem med størrelsesforholdet). Denne metode bliver kaldt fra et andet scripts(DecalWindow.cs, DecalObjectEditor.cs)

#### 2.1.1.3.1 CreateMesh()

Laver en firkantet mesh der gør det muligt at se gameobjectets materiale.

### 2.1.2 BlitTexture2DExt

Bruges ikke mere. Se DecalObject.cs beskrivelse.

Denne klasse er for extension metoder. Kombinere to Texture2D til en Texture2D. (Dette virker ikke helt som vi havde håbet da der er problemet med størrelsesforholdet.)

#### 2.1.2.2.1 Blit()

Her kombinere vi de to Texture2D der skal blive til et. Vi har hoved Texture2D(texture) også det andet(decal) der skal tilføjes på en angivet position(position).

### 2.1.3 DecalObjectEditor

Bruges ikke mere. Se DecalObject.cs beskrivelse

Vi har lavet dette CustomEditor så vi nemt kan flytte decal (CTRL + Klik) også kan vi se et billede af valgte decal plus vi kan ændre flere på en gang.

#### 2.1.3.1.1 OnEnable()

Når scriptet bliver aktiv, finder vi de properties der skal kunne blive ændret i og gemmer dem så vi ikke skal bruge resources på at finde dem hele tiden.

#### 2.1.3.1.2 OnSceneGUI()

Tjekker om man holder ctrl nede og klikker på en overflade, hvis man gør det flytter vi decal der hen.

#### 2.1.3.1.3 OnInspectorGUI()

Her ændrer vi udseende for DecalObject scripet i Inspector vinduet. Vi gør dette i et CustomEditor script så vi kan se billedet af decal.

### 2.1.4 DecalWindow

Bruges ikke mere. Se DecalObject.cs beskrivelse.

Et editor vindue der gør det nemt at oprette nye decals.

#### 2.1.4.1.1 OnGUI()

Udseende for vores vindue.

#### 2.1.4.2.1 ShowWindow()

Åbner vores vindue.

#### 2.1.4.3.1 Button\_CreateDecak\_Click()

Når man klikker på "Add Decal To Scene" knappen laver vi et nyt gameobject og giver den de Component den skal have og kalder .BuildDecal() fra DecalObject.cs

## 2.2 PrefabChanger

### 2.2.1 PrefabChangerWindow

Med dette kan vi sætte mange prefabs ind i en liste hvorefter vi så kan vælge en eller flere gameobjecter også sige at de skal skifte ud med det valgte prefab i listen.

#### 2.2.1.1.1 OnGUI()

Vores vindue udseende

#### 2.2.1.2.1 ShowWindow()

Åbner vores vindue.

#### 2.2.1.3.1 Button\_OnMakeChange()

Apply knap. Her udskifter vi de valgte gameobjecter.

#### 2.2.1.3.2 Button\_OnRemove()

Fjerner valgte option fra listen.

#### 2.2.1.3.3 InsertPrefabOption()

Tilføjer en ny option til listen.

### 2.2.2 PrefabOption

Option box der kan holde på en prefab med apply og remove knap. Har unik id så man kan kende forskel.

#### 2.2.2.2.1 PrefabOption()

Giver vores fields værdier.

#### 2.2.2.2.2 Display()

Udseende for option box.

### 2.2.3 ButtonInfo

En Button der kan udføre et event.

#### 2.2.3.2.1 ButtonInfo()

Giver vores fields værdier.

#### 2.2.3.2.2 Display()

Udseende for knap.

## 2.3 PrefabPlacer

### 2.3.1 PrefabPlacerWindow

Dette er et plugin til Unity som ikke har noget med spillet at gøre. Denne klasse gør at vi kan placere mange prefabs på en engang i stedet for at placer 1 af gangen. Når man holder musen nede bliver der placeret prefabs inden for den radius man har valgt. Man vælger selv prefabs.

#### 2.3.1. 1.1 OnGUI()

Tilføjer tingene der bliver vist i editor siden.

#### 2.3.1. 1.2 OnFocus()

Vi tilføjer et event til sceneview.

#### 2.3.1. 3.1OnSceneView()

Vi opretter en mulighed for at kunne fortryde(undo). Hvis man klikker på et object, begynder vi at placere prefabs.

#### 2.3.1. 3.2 RegisterUndo()

den her metode gør at vi kan lave undo(ctrl + z) og redo(ctrl + y).

#### 2.3.1. 3.3 HasCollider()

Tjekker om gameobjectet og dens child har en collider.

#### 2.3.1. 3.4 BeginInstantiateGameObjects()

Her placere vi de valgte prefabs i cirkler og hvis de er for tæt på hinanden bliver de sprunget over.

#### 2.3.1. 3.5 ValidatePrefabs()

Tjekker listen med prefabs igennem om de har colliders og at de ikke er null.

#### 2.3.1. 3.6 PlacePrefab()

Tjekker om prefabs kan blive placeret der hvor musen er.

#### 2.3.1. 3.7 GetPosition()

Tager center point og tilføjer radius og drejer den rundt om center point, med det angivet degree.

#### 2.3.1. 3.8 GetDegree()

Konvertere vores space variable om til grader så vi kan lave en cirkel.

#### 2.3.1. 3.9 GetBounds()

Finder størrelsen på det valgte gameobject og hvor meget vi kan tilføje for at den er over jorden.

#### 2.3.1. 3.10 ObjectFieldArray<T>()

Array version af EditorGUILayout.ObjectField.

## 2.4 MaterialReplacer

### 2.4.1 MaterialReplacerWindow

Den her klasse gør det muligt for os at udskifte materialer på mange forskellige objecter på samme tid og på en nem måde.

#### 2.4.1.1.1 OnGUI()

Her laver vi layout på vores vindue og fortæller hvad de forskellige ting skal gøre.

#### 2.4.1.2.1 ShowWindow()

Viser vinduet.

#### 2.4.1.2.2 ReplaceMaterial()

Leder efter et material og udskifter den med noget andet material. Man kan vælge at den også skal tjekke child igennem.

#### 2.4.1.3.1 GetMaterial()

Henter material fra gameobject.

#### 2.4.1.3.2 HasMaterial()

Tjekker om gameobjectet har et material.

### 2.4.1.3.3 SetReplaceMaterial()

Leder efter et material og udskifter den med noget andet material.

## 2.5 RandomHouseColor

### 2.5.1 RandomHouseColorParent

Klassen tilføjes til de gameobjecter som skal skifte. farve når man er i CustomEditor og klikker på en knap.

#### 2.5.1. 2.1 ChangeColorByRandom()

Giver gameobjectet en tilfældig farve fra vores liste af farver.

#### 2.5.1. 2.2 ReplaceMaterial()

Leder efter et material og udskifter den med noget andet material. Man kan vælge at den også skal tjekke child igennem.

#### 2.5.1. 3.1 GetMaterial()

Henter material fra gameobject.

#### 2.5.1. 3.2 HasMaterial()

Tjekker om gameobjectet har et material.

#### 2.5.1. 3.3 SetReplaceMaterial()

Leder efter et material og udskifter den med noget andet material.

### 2.5.2 RandomHouseColorEditor

Tilføjer en knap til gameobject som har RandomHouseColorParent component, når man trykke på knappen skifter den farverne.

#### 2.5.2.1.1 OnInsectorGUI()

Tilføjer en knap, når man trykker på den skifter den farverne.