Kode Indexering

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

using UnityEngine;

using Assets.Plugins.Decals.Scripts;

[RequireComponent(typeof(MeshFilter))]

[RequireComponent(typeof(MeshRenderer))]

[ExecuteInEditMode]

public class DecalObject : MonoBehaviour {

#### 2.1.1.0.1 Attributter

public RaycastHit hit;

public Sprite image;

public float offset = 0.01f;

private MeshFilter filter;

private new MeshRenderer renderer; //new fjerner en warning

#### 2.1.1.1.1 OnEnable()

void OnEnable()

{

if (Application.isEditor && !Application.isPlaying)

{

BuildDecal();

}

}

#### 2.1.1.2.1 BuildDecal()

public void BuildDecal()

{

//Hent commponents og hvis den ikke har dem add dem

filter = gameObject.GetComponent<MeshFilter>() != null ? gameObject.GetComponent<MeshFilter>() : gameObject.AddComponent<MeshFilter>();

renderer = gameObject.GetComponent<MeshRenderer>() != null ? gameObject.GetComponent<MeshRenderer>() : gameObject.AddComponent<MeshRenderer>();

//Lav en firkant så vi kan se billedet

filter.mesh = CreateMesh(transform.localScale.x, transform.localScale.y);

// lavet nyt material der har samme størrelse som decal billedet

var tempMaterial = new Material(Shader.Find("Transparent/Diffuse")); //Går den kan være usynlig nogle steder

var croppedTexture = new Texture2D((int)image.rect.width, (int)image.rect.height);

var pixels = image.texture.GetPixels((int)image.textureRect.x,

(int)image.textureRect.y,

(int)image.textureRect.width,

(int)image.textureRect.height);

//Tilføj billede i materialet

croppedTexture.SetPixels(pixels);

croppedTexture.Apply();//decal

//Hvis den er "tæt"(skal klikke på et gameobject) på et gameobject

//vil den tegne på det gameobject materiale

if (hit.collider != null)

{

var o = (Texture2D)hit.collider.GetComponent<Renderer>().material.mainTexture;

o.Blit(croppedTexture, hit.textureCoord);

}

//Giv decal gameobject materialet

//tempMaterial.shader = Shader.Find("Transparent/Diffuse");

tempMaterial.mainTexture = croppedTexture;

renderer.sharedMaterial = tempMaterial;

}

#### 2.1.1.3.1 CreateMesh()

private Mesh CreateMesh(float width, float height)

{

//vertices starter fra midten derfor skal vi kun bruge halvdelen

width = width / 2;

height = height / 2;

Mesh m = new Mesh();

m.name = "Decal";

//definere hjørnerne

m.vertices = new Vector3[] {

new Vector3(width, height, offset),

new Vector3(-width, height,offset),

new Vector3(-width, -height, offset),

new Vector3(width, -height, offset)

};

// hvordan materialet skal sidde

m.uv = new Vector2[] {

new Vector2 (0, 1),

new Vector2 (1, 1),

new Vector2(1, 0),

new Vector2 (0, 0)

};

m.triangles = new int[] { 0, 1, 2, 0, 2, 3 };

m.RecalculateNormals();

return m;

}

### 2.1.2 BlitTexture2DExt

using UnityEngine;

namespace Assets.Plugins.Decals.Scripts

{

public static class BlitTexture2DExt

#### 2.1.2.2.1 Blit()

public static void Blit(this Texture2D texture, Texture2D decal, Vector2 position)

{

//put decal on texture

Color[] decalColors = decal.GetPixels();

int decalWidth = decal.width;

int decalHeight = decal.height;

texture.SetPixels((int)(texture.width \* position.x), (int)(texture.height \* position.y), decalWidth, decalHeight, decalColors);

texture.Apply();

}

### 2.1.3 DecalObjectEditor

using UnityEngine;

using UnityEditor;

[CanEditMultipleObjects]

[CustomEditor(typeof(DecalObject))]

public class DecalObjectEditor : Editor

#### 2.1.3.0.1 Attributter

private SerializedProperty imageProp;

private SerializedProperty offsetProp;

#### 2.1.3.1.1 OnEnable()

void OnEnable()

{

// Setup the SerializedProperties.

imageProp = serializedObject.FindProperty("image");

offsetProp = serializedObject.FindProperty("offset");

}

#### 2.1.3.1.2 OnSceneGUI()

void OnSceneGUI()

{

// Gør at vi ikke vælger nyt object når vi holder ctrl nede og klikker på noget

if (Event.current.control)

HandleUtility.AddDefaultControl(GUIUtility.GetControlID(FocusType.Passive));

//Flyt decal hent til det vi trykket på og vend den rigtigt

if (Event.current.control && Event.current.type == EventType.MouseDown)

{

DecalObject decal = (DecalObject)target;

Ray ray = HandleUtility.GUIPointToWorldRay(Event.current.mousePosition);

RaycastHit hit;

if (Physics.Raycast(ray, out hit, 50))

{

GameObject moveMe = decal.gameObject;

//Kopir valgt decal og flyt den hent til hvor vi trykker.

if (Event.current.shift)

{

moveMe = Instantiate<GameObject>(decal.gameObject);

moveMe.name = decal.name;

}

//Sæt position

moveMe.transform.position = hit.point;

moveMe.transform.forward = hit.normal;

moveMe.GetComponent<DecalObject>().hit = hit;

EditorUtility.SetDirty(moveMe);

}

}

}

#### 2.1.3.1.3 OnInspectorGUI()

public override void OnInspectorGUI()

{

serializedObject.Update();

//DecalObject decal = (DecalObject)target;

//decal.image = (Sprite)EditorGUILayout.ObjectField("Image", decal.image, typeof(Sprite), false);

//decal.offset = EditorGUILayout.Slider("Offset", decal.offset, 0.01f, 0.1f);

imageProp.objectReferenceValue = EditorGUILayout.ObjectField("Image", imageProp.objectReferenceValue, typeof(Sprite), false);

EditorGUILayout.Slider(offsetProp, 0.01f, 0.1f, new GUIContent("Offset"));

serializedObject.ApplyModifiedProperties();

if (GUI.changed)

{

foreach (var item in serializedObject.targetObjects)

{

((DecalObject)item).BuildDecal();

EditorUtility.SetDirty(((DecalObject)item));

}

}

}

### 2.1.4 DecalWindow

using UnityEngine;

using UnityEditor;

public class DecalWindow : EditorWindow

#### 2.1.4.0.1 Attributter

private string btnCreate = "Add Decal To Scene";

private Sprite decal;

#### 2.1.4.1.1 OnGUI()

void OnGUI()

{

float marginLeft = this.position.width / 2 - 100 / 2;

decal = (Sprite)EditorGUI.ObjectField(new Rect(marginLeft, 5, 100, 100), decal, typeof(Sprite), true);

GUILayout.BeginArea(new Rect(5, 125, this.position.width - 10, 32));

if (GUILayout.Button(btnCreate))

Button\_CreateDecal\_Click();

GUILayout.EndArea();

}

#### 2.1.4.2.1 ShowWindow()

[MenuItem("Window/Decals")]

public static void ShowWindow()

{

GetWindow<DecalWindow>().Show();

}

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

private void Button\_CreateDecal\_Click()

{

GameObject go = new GameObject("Decal");

go.AddComponent<MeshFilter>();

go.AddComponent<MeshRenderer>();

var decalObj = go.GetComponent<DecalObject>() ?? go.AddComponent<DecalObject>();

decalObj.image = decal;

decalObj.BuildDecal();

decalObj.offset = 1f;

Selection.activeGameObject = go;

}

## 2.2 PrefabChanger

### 2.2.1 PrefabChangerWindow

using System.Collections.Generic;

using System.Linq;

using UnityEngine;

using UnityEditor;

using Assets.Plugins.PrefabChanger.Scripts;

public class PrefabChangerWindow : EditorWindow {

#### 2.2.1.0.1 Attributter

private List<PrefabOption> prefabList = new List<PrefabOption>();

private GameObject prefab;

private Vector2 imageSize = new Vector2(100, 100);

private Vector2 scrollPosition = Vector2.zero;

#### 2.2.1.1.1 OnGUI()

void OnGUI()

{

// Skal have en tom plads tilovers

if (prefabList.Count == 0 || prefabList[prefabList.Count - 1].prefab != null)

{

InsertPrefabOption();

}

// Knap til at fjerne alle options

var clearBtn = new ButtonInfo(-1, "Reset", new Vector2(5, 5), (s) => { prefabList.Clear(); });

clearBtn.Display(position.width);

//information der skal ses i vinduet

EditorGUI.HelpBox(new Rect(5,27, position.width- 10, 30), "Click 'Apply' to change selected Gameobjects.", MessageType.Info);

EditorGUI.HelpBox(new Rect(5, 59, position.width - 10, 30), "Components will also be changed", MessageType.Warning);

//G�r at de st�r lige efter hinanden ogs� hvis man fjerner en i midten

for (int i = 0; i < prefabList.Count; i++)

{

prefabList[i].margin = new Vector2(5, 100 \* i);

}

//Start scroll box

scrollPosition = GUI.BeginScrollView(new Rect(0, 90, position.width, position.height-90), scrollPosition, new Rect(0, 0, 220, 100\* prefabList.Count));

// Lav en midlertidig liste at da den rigtige kan midste sine option ved at de bliver fjernet

// hvilket kan f� et for loop til at g� i stykker

var tempPrefabsList = new List<PrefabOption>();

tempPrefabsList.AddRange(prefabList);

foreach (var item in tempPrefabsList)

{

item.Display(position.width);

}

GUI.EndScrollView();

}

#### 2.2.1.2.1 ShowWindow()

[MenuItem("Window/Prefab Changer")]

public static void ShowWindow()

{

GetWindow<PrefabChangerWindow>("Prefab Changer").Show();

}

#### 2.2.1.3.1 Button\_OnMakeChange()

private void Button\_OnMakeChange(object sender)

{

//v�r sikker p� at denne knap ikke er tom(prefab)

if (prefabList.Count(m => m.index == ((ButtonInfo)sender).index) > 0)

{

//Hent prefab

GameObject replacement = prefabList.First(m => m.index == ((ButtonInfo)sender).index).prefab;

//Selection.activeGameObject = GameObject.Instantiate<GameObject>(replacement);

//Udskift alle gameobjecterne

for (int i = Selection.gameObjects.Length - 1; i >= 0; i--)

{

//Gamle v�rdier der skal tilf�jes til det nye gameobject(prefab)

var oldTransform = Selection.gameObjects[i].transform;

var oldName = Selection.gameObjects[i].name;

var tfParent = oldTransform.parent;

var tfPosition = oldTransform.position;

var tfRotation = oldTransform.rotation;

var tfLocalScale = oldTransform.localScale;

// var oldComponents = Selection.gameObjects[i].GetComponents<Component>();

// Selection.gameObjects[i] = GameObject.Instantiate<GameObject>(replacement);

//Fjern det gamle gameobject

GameObject.DestroyImmediate(Selection.gameObjects[i]);

//Opret nyt gameobject

var go = GameObject.Instantiate<GameObject>(replacement);

//Giv det nye gameobject v�rdier fra det gamle

go.name = oldName;

go.transform.parent = tfParent;

go.transform.position = tfPosition;

go.transform.rotation = tfRotation;

go.transform.localScale = tfLocalScale;

}

}

}

#### 2.2.1.3.2 Button\_OnRemove()

private void Button\_OnRemove(object sender)

{

if (prefabList.Count(m => m.index == ((ButtonInfo)sender).index) > 0)

{

prefabList.Remove(prefabList.First(m => m.index == ((ButtonInfo)sender).index));

}

}

#### 2.2.1.3.3 InsertPrefabOption()

private void InsertPrefabOption()

{

prefabList.Add(new PrefabOption(Button\_OnMakeChange, Button\_OnRemove));

}

### 2.2.2 PrefabOption

using UnityEngine;

using UnityEditor;

namespace Assets.Plugins.PrefabChanger.Scripts

{

public class PrefabOption

#### 2.2.2.0.1 Attributter

private static int indexCounter = 0;

public int index;

public GameObject prefab;

private ButtonInfo makeChange;

private ButtonInfo remove;

public Vector2 margin;

#### 2.2.2.2.1 PrefabOption()

public PrefabOption(ButtonInfo.OnClickDelegate onMakeChange = null, ButtonInfo.OnClickDelegate onRemove = null)

{

index = ++indexCounter;

this.margin = Vector2.zero;

this.prefab = null;

this.makeChange = new ButtonInfo(index, "Apply", new Vector2(115, 55), onMakeChange);

this.remove = new ButtonInfo(index, "Remove", new Vector2(115, 80), onRemove);

}

#### 2.2.2.2.2 Display()

public void Display(float width)

{

//Gem det hele i en box så vi nemt at flytte på alle elementerne

float widthWithoutMargin = width - margin.x \* 2;

GUI.BeginGroup(new Rect(margin, new Vector2(widthWithoutMargin, 100)));

//Gør at man kan vælge prefab

prefab = (GameObject)EditorGUI.ObjectField(new Rect(115, 5, widthWithoutMargin - 115, 18), prefab, typeof(GameObject), true);

//Vis billedet af prefab hvis den har billedet

// ellers hvis en hvid box

var picRect = new Rect(5, 5, 100, 100);

if (prefab != null)

{

EditorGUI.DrawPreviewTexture(picRect, AssetPreview.GetAssetPreview(prefab));

}

else

{

EditorGUI.DrawRect(picRect, Color.white); ;

}

//this.makeChange.isEnabled = prefab != null;

//this.remove.isEnabled = prefab != null;

//Knapperne

this.makeChange.Display(widthWithoutMargin - 115);

this.remove.Display(widthWithoutMargin - 115);

GUI.EndGroup();

}

### 2.2.3 ButtonInfo

using UnityEngine;

namespace Assets.Plugins.PrefabChanger.Scripts

{

public class ButtonInfo

#### 2.2.3.0.1 Attributter

public delegate void OnClickDelegate(object sender);

public int index;

public bool isEnabled;

public string title;

public Vector2 margin;

public OnClickDelegate onClick;

#### 2.2.3.2.1 ButtonInfo()

public ButtonInfo(int id, string title, Vector2 margin, OnClickDelegate onClick)

{

this.index = id;

this.title = title;

this.margin = margin;

this.onClick = onClick;

this.isEnabled = true;

}

#### 2.2.3.2.2 Display()

public void Display(float width)

{

if (isEnabled)

{

if (GUI.Button(new Rect(margin, new Vector2(width - 10, 20)), title))

{

onClick(this);

}

}

}

## 2.3 PrefabPlacer

### 2.3.1 PrefabPlacerWindow

using System.Collections.Generic;

using System.Linq;

using UnityEditor;

using UnityEngine;

public class PrefabsPlacerWindow : EditorWindow {

#### 2.3.1.0.1 Attributter

private int radius = 25;

private float space = 4f;

private bool isEnabled = false;

private bool isMouseDown = false;

private GameObject parent;

private GameObject ground;

private GameObject ground2;

private List<Bounds?> prefabsBounds = new List<Bounds?>();

private List<Vector3?> prefabsPlus = new List<Vector3?>();

private int prefabsSize = 1;

private GameObject[] prefabs;

private GameObject undoParent;

#### 2.3.1. 1.1 OnGUI()

void OnGUI()

{

// Laver mellemrum i editor

EditorGUILayout.Separator();

EditorGUILayout.Separator();

EditorGUILayout.Separator();

EditorGUILayout.Separator();

EditorGUILayout.Separator();

EditorGUILayout.Separator();

EditorGUILayout.Separator();

// Checkbox som ændre på en bool som hedder isEnabled

isEnabled = EditorGUILayout.Toggle("Is Enabled", isEnabled);

EditorGUILayout.Separator();

EditorGUILayout.Separator();

radius = EditorGUILayout.IntSlider("Radius", radius, 5, 100);

space = EditorGUILayout.Slider("Space Between Prefabs", space, 4f, radius);

EditorGUILayout.Separator();

EditorGUILayout.Separator();

ground2 = (GameObject)EditorGUILayout.ObjectField("Ground", ground2, typeof(GameObject), true);

parent = (GameObject)EditorGUILayout.ObjectField("Container", parent, typeof(GameObject), true);

EditorGUILayout.Separator();

prefabsSize = EditorGUILayout.IntSlider("Number Of Prefabs", prefabsSize, 1, 25);

prefabs = ObjectFieldArray<GameObject>("Prefab", prefabsSize, prefabs);

EditorGUILayout.Separator();

EditorGUILayout.Separator();

EditorGUILayout.HelpBox("Prefabs need a collider", MessageType.Info);

if (GUI.Button(new Rect(5, 5, 100, 32), "Reset Ground"))

{

ground2 = null;

}

if (GUI.changed)

{

//ground = null;

ValidatePrefabs();

}

}

#### 2.3.1. 1.2 OnFocus()

void OnFocus()

{

SceneView.onSceneGUIDelegate += OnSceneView;

}

#### 2.3.1.2.1 ShowWindow

[MenuItem("Window/Prefabs Placer")]

public static void ShowWindow()

{

GetWindow<PrefabsPlacerWindow>().Show();

}

#### 2.3.1. 3.1OnSceneView()

private void OnSceneView(SceneView scene)

{

if (isEnabled)

{

HandleUtility.AddDefaultControl(GUIUtility.GetControlID(FocusType.Passive));

if (Event.current.button == 0 && Event.current.type == EventType.MouseDown && !isMouseDown)

{

RegisterUndo();

isMouseDown = true;

}

if ((Event.current.button == 0 && Event.current.type == EventType.MouseUp) || Event.current.type == EventType.Ignore)

{

ground = null;

isMouseDown = false;

if (undoParent != null && undoParent.transform.childCount == 0)

{

DestroyImmediate(undoParent);

}

}

RaycastHit hit;

Ray ray = HandleUtility.GUIPointToWorldRay(Event.current.mousePosition);

if (Physics.Raycast(ray, out hit))

{

Handles.DrawWireDisc(hit.point, Vector3.up, radius);

HandleUtility.Repaint();

if (isMouseDown)

{

if (ground == null)

{

ground = hit.collider.gameObject;

}

BegindInstantiateGameObjects(hit.point);

}

}

}

}

#### 2.3.1. 3.2 RegisterUndo()

private void RegisterUndo()

{

undoParent = new GameObject("Prefabs Group");

if (parent == null)

{

parent = new GameObject("Prefabs");

Undo.RegisterCreatedObjectUndo(parent, "Undo Prefabs");

}

else

{

Undo.RegisterCreatedObjectUndo(undoParent, "Undo Prefabs Group");

}

undoParent.transform.parent = parent.transform;

}

#### 2.3.1. 3.3 HasCollider()

private bool HasCollider(GameObject item)

{

if (item.GetComponent<Collider>() != null)

return true;

// Tjekker om der er et child i objectet som har en collider

for (int i = 0; i < item.transform.childCount; i++)

if (HasCollider(item.transform.GetChild(i).gameObject))

return true;

return false;

}

#### 2.3.1. 3.4 BeginInstantiateGameObjects()

private void BegindInstantiateGameObjects(Vector3 center)

{

// Vi checker variablerne igennem for null

var prefabList = prefabs.Where(m => m != null).ToArray();

var plusList = prefabsPlus.Where(m => m != null).ToArray();

var boundsList = prefabsBounds.Where(m => m != null).ToArray();

int id = Random.Range(0, prefabList.Length);

Vector2 g = new Vector2(center.x, center.z);

// sætter et prefab

PlacePrefab(prefabList[id], boundsList[id].Value, center, plusList[id].Value);

float radius = space;

// checker på om de er for tæt på hinanden

while (radius < this.radius)

{

int? degrees = GetDegree(center, radius, space);

if (degrees.HasValue)

{

for (int i = 0; i < 360; i += degrees.Value)

{

id = Random.Range(0, prefabList.Length);

PlacePrefab(prefabList[id], boundsList[id].Value, GetPosition(center, radius, i), plusList[id].Value);

}

}

radius += space;

}

}

#### 2.3.1. 3.5 ValidatePrefabs()

private void ValidatePrefabs()

{

prefabsPlus.Clear();

prefabsBounds.Clear();

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

{

GameObject item = prefabs[i];

if (item != null && HasCollider(item))

{

Vector3 plus;

prefabsBounds.Add(GetBounds(item, out plus, null));

prefabsPlus.Add(plus);

}

else

{

prefabs[i] = null;

prefabsBounds.Add(null);

prefabsPlus.Add(null);

}

}

}

#### 2.3.1. 3.6 PlacePrefab()

private void PlacePrefab(GameObject prefab, Bounds bounds, Vector3 position, Vector3 plus)

{

// checker hvor prefabs skal sætte på.

GameObject mainGround = ground2 != null ? ground2 : ground;

RaycastHit hit;

bool canCreate = true;

float maxSize = bounds.size.x > bounds.size.z ? bounds.size.x : bounds.size.z;

if (Physics.Raycast(position + Vector3.up, Vector3.down, out hit, 50))

{

position.y = hit.point.y;

canCreate = hit.collider.gameObject.GetInstanceID() == mainGround.GetInstanceID();

}

else if (Physics.Raycast(position + Vector3.down, Vector3.up, out hit, 50))

{

position.y = hit.point.y;

canCreate = hit.collider.gameObject.GetInstanceID() == mainGround.GetInstanceID();

}

else

{

canCreate = false;

}

if (canCreate)

{

Collider[] cols = Physics.OverlapSphere(position, space - (maxSize / 2)/\*-2\*/);

foreach (var item in cols)

{

if (item.gameObject.GetInstanceID() != mainGround.GetInstanceID())

{

canCreate = false;

break;

}

}

}

if (canCreate)

{

GameObject go2 = Instantiate(prefab, position + plus, Quaternion.identity);

var rotation = go2.transform.rotation;

rotation \*= Quaternion.Euler(0, Random.Range(0, 359), 0); // this adds a 90 degrees Y rotation

go2.transform.rotation = rotation;

go2.transform.parent = undoParent.transform;

}

}

#### 2.3.1. 3.7 GetPosition()

private Vector3 GetPosition(Vector3 center, float radius, float degree)

{

float a = degree \* Mathf.PI / 180;

float x = center.x + radius \* Mathf.Cos(a);

float z = center.z + radius \* Mathf.Sin(a);

return new Vector3(x, center.y, z);

}

#### 2.3.1. 3.8 GetDegree()

private int? GetDegree(Vector3 center, float radius, float space)

{

Vector3 start = GetPosition(center, radius, 0);

for (int i = 0; i < 360; i++)

{

Vector3 check = GetPosition(center, radius, i);

if (Vector3.Distance(start, check) >= space)

{

return i;

}

}

return null;

}

#### 2.3.1. 3.9 GetBounds()

private Bounds? GetBounds(GameObject main, out Vector3 plus, Vector3? worldPosition)

{

bool isFirst = !worldPosition.HasValue;

Vector3 position = main.transform.position;

worldPosition = position;

Vector3 max = new Vector3();

Vector3 min = new Vector3();

Bounds? bounds = null;

Renderer renderer = main.GetComponent<Renderer>();

bool hasRenderer = renderer != null;

bool isMaxMinSet = false;

if (hasRenderer)

{

max = renderer.bounds.max;

min = renderer.bounds.min;

isMaxMinSet = true;

}

for (int i = 0; i < main.transform.childCount; i++)

{

Vector3 unUse;

Bounds? childBounds = GetBounds(main.transform.GetChild(i).gameObject, out unUse, worldPosition);

if (childBounds.HasValue)

{

if (!hasRenderer && !isMaxMinSet)

{

max = childBounds.Value.max;

min = childBounds.Value.min;

isMaxMinSet = true;

}

if (childBounds.Value.max.x > max.x) max.x = childBounds.Value.max.x;

if (childBounds.Value.max.y > max.y) max.y = childBounds.Value.max.y;

if (childBounds.Value.max.z > max.z) max.z = childBounds.Value.max.z;

if (childBounds.Value.min.x < min.x) min.x = childBounds.Value.min.x;

if (childBounds.Value.min.y < min.y) min.y = childBounds.Value.min.y;

if (childBounds.Value.min.z < min.z) min.z = childBounds.Value.min.z;

}

}

if (isMaxMinSet)

{

Bounds defineBounds = new Bounds();

defineBounds.max = max;

defineBounds.min = min;

bounds = defineBounds;

}

if (isFirst)

{

plus = new Vector3(0, position.y - bounds.Value.min.y, 0);

}

else

{

plus = Vector3.zero;

}

return bounds;

}

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

public static T[] ObjectFieldArray<T>(string label, int size, T[] objs) where T : UnityEngine.Object

{

if (size > 0)

{

if (objs == null)

objs = new T[size];

if (size != objs.Length)//resize the array

{

//Assuming array isn't enormous in size, just copying the entire array each time should be fine

T[] newArray = new T[size];

for (int i = 0; i < size; i++)//Copy whatever we can

{

newArray[i] = i < objs.Length ? objs[i] : null;

}

objs = new T[size];

for (int i = 0; i < size; i++)//Copy back whatever we can

{

objs[i] = newArray[i];

}

}

EditorGUI.indentLevel++;

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

{

objs[i] = EditorGUILayout.ObjectField(label + " " + i, objs[i], typeof(T), true) as T;

}

EditorGUI.indentLevel--;

}

else

{

return null; //Don't return anything

}

return objs;

}

## 2.4 MaterialReplacer

### 2.4.1 MaterialReplacerWindow

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEditor;

public class MaterialReplacerWindow : EditorWindow {

#### 2.4.1.0.1 Attributter

private List<Material> m\_find;

private List<Material> m\_replace;

private List<int> m\_id;

private Vector2 scrollPos = new Vector2(0,0);

bool child;

int count = 1;

#### 2.4.1.1.1 OnGUI()

void OnGUI()

{

// Laver et lille mellemrum

EditorGUILayout.Separator();

// Er en checkbox som checker om vi skal have children med

child = GUILayout.Toggle(child, "Child");

scrollPos = EditorGUILayout.BeginScrollView(scrollPos);

for (int a = 0; a < m\_find.Count; a++)

{

//if (a == m\_find.Count)

//{

// m\_find.Add(null);

// m\_replace.Add(null);

//}

EditorGUILayout.Separator();

EditorGUILayout.Separator();

m\_find[a] = (Material)EditorGUILayout.ObjectField("Find", m\_find[a], typeof(Material), false);

m\_replace[a] = (Material)EditorGUILayout.ObjectField("Replace", m\_replace[a], typeof(Material), false);

EditorGUILayout.Separator();

EditorGUILayout.Separator();

if (GUILayout.Button("Switch"))

{

Material temp = m\_find[a];

m\_find[a] = m\_replace[a];

m\_replace[a] = temp;

}

if (GUILayout.Button("Apply") && m\_find[a] != null && m\_replace[a] != null)

{

foreach (var go in Selection.gameObjects)

{

ReplaceMaterial(go, m\_find[a], m\_replace[a], child);

}

}

EditorGUILayout.Separator();

if (GUILayout.Button("Remove"))

{

count--;

m\_find.RemoveAt(a);

m\_replace.RemoveAt(a);

a--;

continue;

}

}

EditorGUILayout.EndScrollView();

EditorGUILayout.Separator();

EditorGUILayout.Separator();

if (GUILayout.Button("Add"))

{

m\_find.Add(null);

m\_replace.Add(null);

}

}

#### 2.4.1.2.1 ShowWindow()

[MenuItem("Window/Material Replacer")]

public static void ShownWindow()

{

GetWindow<MaterialReplacerWindow>().Show();

}

#### 2.4.1.2.2 ReplaceMaterial()

public static void ReplaceMaterial(GameObject go, Material find, Material replace, bool includeChild = true)

{

SetReplaceMaterial(go, find, replace);

if (includeChild)

{

for (int i = 0; i < go.transform.childCount; i++)

{

GameObject child = go.transform.GetChild(i).gameObject;

ReplaceMaterial(child, find, replace, includeChild);

}

}

}

#### 2.4.1.3.1 GetMaterial()

private static Material GetMaterial(GameObject gameObject)

{

if (gameObject.GetComponent<Renderer>() != null)

{

return gameObject.GetComponent<Renderer>().sharedMaterial;

}

return null;

}

#### 2.4.1.3.2 HasMaterial()

private static bool HasMaterial(GameObject go, Material material)

{

return GetMaterial(go) == material;

}

### 2.4.1.3.3 SetReplaceMaterial()

private static void SetReplaceMaterial(GameObject go, Material find, Material replace)

{

if (HasMaterial(go, find))

{

go.GetComponent<Renderer>().sharedMaterial = replace;

}

}

## 2.5 RandomHouseColor

### 2.5.1 RandomHouseColorParent

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[ExecuteInEditMode]

public class RandomHouseColorParent : MonoBehaviour {

#### 2.5.1.0.1 Attributter

public Material[] m\_colors;

#### 2.5.1. 2.1 ChangeColorByRandom()

public void ChangeColorByRandom()

{

if (m\_colors.Length == 0)

{

return;

}

GameObject[] houses = new GameObject[transform.childCount];

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

{

houses[i] = transform.GetChild(i).gameObject;

}

foreach (var house in houses)

{

Material replace = m\_colors[Random.Range(0, m\_colors.Length)];

foreach (var find in m\_colors)

{

ReplaceMaterial(house, find, replace);

}

//Debug.Log(house.transform.position);

//var hits = Physics.RaycastAll(house.transform.position, -house.transform.right, 2);

//if (hits.Length > 0)

//{

// Debug.Log("true");

// foreach (var item in hits)

// {

// Debug.Log(item.collider.transform.position);

// }

// break;

//}

}

}

#### 2.5.1. 2.2 ReplaceMaterial()

public static void ReplaceMaterial(GameObject go, Material find, Material replace, bool includeChild = true)

{

SetReplaceMaterial(go, find, replace);

if (includeChild)

{

for (int i = 0; i < go.transform.childCount; i++)

{

GameObject child = go.transform.GetChild(i).gameObject;

ReplaceMaterial(child, find, replace);

}

}

}

#### 2.5.1. 3.1 GetMaterial()

private static Material GetMaterial(GameObject gameObject)

{

if (gameObject.GetComponent<Renderer>() != null)

{

return gameObject.GetComponent<Renderer>().sharedMaterial;

}

return null;

}

#### 2.5.1. 3.2 HasMaterial()

private static bool HasMaterial(GameObject go, Material material)

{

return GetMaterial(go) == material;

}

#### 2.5.1. 3.3 SetReplaceMaterial()

private static void SetReplaceMaterial(GameObject go, Material find, Material replace)

{

if (HasMaterial(go, find))

{

go.GetComponent<Renderer>().sharedMaterial = replace;

}

}

### 2.5.2 RandomHouseColorEditor

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEditor;

[CustomEditor(typeof(RandomHouseColorParent))]

public class RandomHouseColorEditor : Editor {

#### 2.5.2.1.1 OnInsectorGUI()

public override void OnInspectorGUI()

{

var controller = target as RandomHouseColorParent;

EditorGUIUtility.LookLikeInspector();

SerializedProperty tps = serializedObject.FindProperty("m\_colors");

EditorGUI.BeginChangeCheck();

EditorGUILayout.PropertyField(tps, true);

if (EditorGUI.EndChangeCheck())

serializedObject.ApplyModifiedProperties();

EditorGUIUtility.LookLikeControls();

//((RandomHouseColorParent)target).m\_colors = (Material[])EditorGUILayout.ObjectField("", ((RandomHouseColorParent)target).m\_colors, typeof(Material[]), false);

if (GUILayout.Button("I am a button"))

{

Debug.Log("hej");

((RandomHouseColorParent)target).ChangeColorByRandom();

}

}

# 3 Shader

## 3.1 Decal Advanced

Shader "Custom/Decal Advanced"{

Properties {

//Farve overlay

\_ColorMain ("Color Main", Color) = (1,1,1,0.5)

\_Color01 ("Color Decal 01", Color) = (1,1,1,0)

\_Color02("Color Decal 02", Color) = (1,1,1,0)

\_Color03("Color Decal 03", Color) = (1,1,1,0)

\_Color04("Color Decal 04", Color) = (1,1,1,0)

\_Color05("Color Decal 05", Color) = (1,1,1,0)

\_Color06("Color Decal 06", Color) = (1,1,1,0)

\_Color07("Color Decal 07", Color) = (1,1,1,0)

\_Color08("Color Decal 08", Color) = (1,1,1,0)

\_Color09("Color Decal 09", Color) = (1,1,1,0)

\_Color10("Color Decal 10", Color) = (1,1,1,0)

//Billedet

\_MainTex ("Main", 2D) = "white" { }

\_DecalTex01 ("Decal 01", 2D) = "white" { }

\_DecalTex02("Decal 02", 2D) = "white" { }

\_DecalTex03("Decal 03", 2D) = "white" { }

\_DecalTex04("Decal 04", 2D) = "white" { }

\_DecalTex05("Decal 05", 2D) = "white" { }

\_DecalTex06("Decal 06", 2D) = "white" { }

\_DecalTex07("Decal 07", 2D) = "white" { }

\_DecalTex08("Decal 08", 2D) = "white" { }

\_DecalTex09("Decal 09", 2D) = "white" { }

\_DecalTex10 ("Decal 10", 2D) = "white" { }

}

SubShader {

Pass {

CGPROGRAM //Shader Start, Vertex Shader named vert, Fragment shader named frag

#pragma vertex vert

#pragma fragment frag

#include "UnityCG.cginc"

//Link properties to the shader

float4 \_ColorMain;

float4 \_Color01;

float4 \_Color02;

float4 \_Color03;

float4 \_Color04;

float4 \_Color05;

float4 \_Color06;

float4 \_Color07;

float4 \_Color08;

float4 \_Color09;

float4 \_Color10;

sampler2D \_MainTex;

sampler2D \_DecalTex01;

sampler2D \_DecalTex02;

sampler2D \_DecalTex03;

sampler2D \_DecalTex04;

sampler2D \_DecalTex05;

sampler2D \_DecalTex06;

sampler2D \_DecalTex07;

sampler2D \_DecalTex08;

sampler2D \_DecalTex09;

sampler2D \_DecalTex10;

///

/// Noget med hvor billederne skal side

///

struct v2f

{

float4 pos : SV\_POSITION;

float2 uv : TEXCOORD0;

float2 uv1 : TEXCOORD1;

float2 uv2 : TEXCOORD2;

float2 uv3 : TEXCOORD3;

float2 uv4 : TEXCOORD4;

float2 uv5 : TEXCOORD5;

float2 uv6 : TEXCOORD6;

float2 uv7 : TEXCOORD7;

float2 uv8 : TEXCOORD8;

float2 uv9 : TEXCOORD9;

float2 uv10 : TEXCOORD10;

};

///

/// Noget med hvor billederne skal side

/// Hvis disse ikke er der bliver billederne ikke flyttet

///

float4 \_MainTex\_ST; //?? skal være der

float4 \_DecalTex01\_ST; //?? skal være der

float4 \_DecalTex02\_ST; //?? skal være der

float4 \_DecalTex03\_ST; //?? skal være der

float4 \_DecalTex04\_ST; //?? skal være der

float4 \_DecalTex05\_ST; //?? skal være der

float4 \_DecalTex06\_ST; //?? skal være der

float4 \_DecalTex07\_ST; //?? skal være der

float4 \_DecalTex08\_ST; //?? skal være der

float4 \_DecalTex09\_ST; //?? skal være der

float4 \_DecalTex10\_ST; //?? skal være der

///

/// Giver billederne deres plads/Definere det

///

v2f vert (appdata\_base v)

{

v2f o;

o.pos = UnityObjectToClipPos (v.vertex); //Transform the vertex position

o.uv = TRANSFORM\_TEX (v.texcoord, \_MainTex); //Prepare the vertex uv

o.uv1 = TRANSFORM\_TEX (v.texcoord, \_DecalTex01); //Prepare the vertex uv

o.uv2 = TRANSFORM\_TEX(v.texcoord, \_DecalTex02); //Prepare the vertex uv

o.uv3 = TRANSFORM\_TEX(v.texcoord, \_DecalTex03); //Prepare the vertex uv

o.uv4 = TRANSFORM\_TEX(v.texcoord, \_DecalTex04); //Prepare the vertex uv

o.uv5 = TRANSFORM\_TEX(v.texcoord, \_DecalTex05); //Prepare the vertex uv

o.uv6 = TRANSFORM\_TEX(v.texcoord, \_DecalTex06); //Prepare the vertex uv

o.uv7 = TRANSFORM\_TEX(v.texcoord, \_DecalTex07); //Prepare the vertex uv

o.uv8 = TRANSFORM\_TEX(v.texcoord, \_DecalTex08); //Prepare the vertex uv

o.uv9 = TRANSFORM\_TEX(v.texcoord, \_DecalTex09); //Prepare the vertex uv

o.uv10 = TRANSFORM\_TEX(v.texcoord, \_DecalTex10); //Prepare the vertex uv

return o;

}

///

/// Her kombinere vi billederne og farverne

///

half4 frag (v2f i) : COLOR

{

//Hent billederne

float4 texcol = tex2D (\_MainTex, i.uv); //base texture

float4 deccol01 = tex2D (\_DecalTex01, i.uv1); //decal texture

float4 deccol02 = tex2D(\_DecalTex02, i.uv2); //decal texture

float4 deccol03 = tex2D(\_DecalTex03, i.uv3); //decal texture

float4 deccol04 = tex2D(\_DecalTex04, i.uv4); //decal texture

float4 deccol05 = tex2D(\_DecalTex05, i.uv5); //decal texture

float4 deccol06 = tex2D(\_DecalTex06, i.uv6); //decal texture

float4 deccol07 = tex2D(\_DecalTex07, i.uv7); //decal texture

float4 deccol08 = tex2D(\_DecalTex08, i.uv8); //decal texture

float4 deccol09 = tex2D(\_DecalTex09, i.uv9); //decal texture

float4 deccol10 = tex2D (\_DecalTex10, i.uv10); //decal texture

//Giver billederne farve

float4 temp00 = \_ColorMain \* \_ColorMain.a \* texcol;

float4 temp01 = \_Color01 \* \_Color01.a \* deccol01;

float4 temp02 = \_Color02 \* \_Color02.a \* deccol02;

float4 temp03 = \_Color03 \* \_Color03.a \* deccol03;

float4 temp04 = \_Color04 \* \_Color04.a \* deccol04;

float4 temp05 = \_Color05 \* \_Color05.a \* deccol05;

float4 temp06 = \_Color06 \* \_Color06.a \* deccol06;

float4 temp07 = \_Color07 \* \_Color07.a \* deccol07;

float4 temp08 = \_Color08 \* \_Color08.a \* deccol08;

float4 temp09 = \_Color09 \* \_Color09.a \* deccol09;

float4 temp10 = \_Color10 \* \_Color10.a \* deccol10;

//Kombinere billederne

temp01 = lerp(temp01, temp02, temp02.a);

temp01 = lerp(temp01, temp03, temp03.a);

temp01 = lerp(temp01, temp04, temp04.a);

temp01 = lerp(temp01, temp05, temp05.a);

temp01 = lerp(temp01, temp06, temp06.a);

temp01 = lerp(temp01, temp07, temp07.a);

temp01 = lerp(temp01, temp08, temp08.a);

temp01 = lerp(temp01, temp09, temp09.a);

temp01 = lerp(temp01, temp10, temp10.a);

return lerp(temp00, temp01, temp01.a);

}

ENDCG //Shader End

}

}

}