LAMPIRAN KODING

Script untuk mengganti ke halaman berikut nya dan keluar dari aplikasi

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ChaneScene : MonoBehaviour {

        public void chanemenuscene(string scenename)
    {
        Application.LoadLevel(scenename);

    }

    public void keluar()
    {
        Application.Quit();
    }
}
```

Script untuk tombol Next and Back pada Scene Bantuan

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BantuanNextPage : MonoBehaviour {

    public int indexPage = 0;

    public GameObject[] Scorlls;

        public void NextPage()
    {
        if (indexPage != Scorlls.Length - 1)
        {
            SetPageActive(Scorlls[indexPage]);
            indexPage++;
            SetPageActive(Scorlls[indexPage]);
        }
    }

    public void PrevPage()
    {
        if (indexPage != 0)
        {
            SetPageActive(Scorlls[indexPage]);
            indexPage--;
```

```
            SetPageActive(Scorlls[indexPage]);
        }
    }

    void SetPageActive(GameObject go)
    {
        go.SetActive(!go.activeSelf);
    }
}
```

Script untuk membuat camera di dalam aplikasi mengikuti gerak gambar 3D

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class jj : MonoBehaviour
{

    Camera cam;

    public Transform target;

    [Header("UI Component")]
    public Text sensityText;
    public Slider slider;

    Vector3 firstPos;
    Vector3 secondPos;

    Vector3 deltaPos;

    float yForward;
    float xRight;

    public float zoomSpeed;

    public float sensity = 5f;

    void Start()
    {
        sensityText.text = sensity.ToString();
        slider.value = sensity;

        cam = GetComponent<Camera>();
    }

    public void SensityChange()
    {
        sensity = slider.value;
        sensityText.text = slider.value.ToString();
    }

    void Update()
    {
        AndroidController();
```

```
        AndroidZoom();
        MoveCamera();
    }

    void AndroidZoom()
    {
        if (Input.touchCount == 2)
        {
            Touch touch0 = Input.GetTouch(0);
            Touch touch1 = Input.GetTouch(1);

            Vector2 touch0_PrevPos = touch0.position - touch0.deltaPosition;
            Vector2 touch1_PrevPos = touch1.position - touch1.deltaPosition;

            float prevTouch_DeltaMag = (touch0_PrevPos -
touch1_PrevPos).magnitude;
            float touch_DeltaMag = (touch0.position -
touch1.position).magnitude;

            float deltaMag_Diff = prevTouch_DeltaMag - touch_DeltaMag;

            cam.fieldOfView += deltaMag_Diff * zoomSpeed;
            cam.fieldOfView = Mathf.Clamp(cam.fieldOfView, .1f, 179.9f);
        }
    }

    void AndroidController()
    {
        if (Input.touchCount > 0)
        {
            if (Input.GetTouch(0).phase == TouchPhase.Began)
            {
                firstPos = Input.GetTouch(0).position;
            }

            if (Input.GetTouch(0).phase == TouchPhase.Moved)
            {
                secondPos = Input.GetTouch(0).position;
                deltaPos = (firstPos - secondPos).normalized;

                transform.Translate(deltaPos * sensity * Time.deltaTime,
Space.Self);
                transform.LookAt(target);
            }
        }
    }

    void MoveCamera()
    {
        if (Input.GetMouseButtonDown(0))
        {
            firstPos = Input.mousePosition;
        }

        if (Input.GetMouseButton(0))
        {
            secondPos = Input.mousePosition;
            deltaPos = (firstPos - secondPos).normalized;
```

```
            transform.Translate(deltaPos * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("w"))
        {
            transform.Translate(Vector3.up * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("s"))
        {
            transform.Translate(Vector3.down * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("d"))
        {
            transform.Translate(Vector3.right * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("a"))
        {
            transform.Translate(Vector3.left * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("q"))
        {
            transform.Translate(Vector3.forward * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("e"))
        {
            transform.Translate(Vector3.back * sensity * Time.deltaTime,
Space.Self);
        }

        transform.LookAt(target);
    }

    void LookTarget()
    {
        Vector3 direction = target.position - transform.position;

        Quaternion lookRotation = Quaternion.LookRotation(direction);
        Vector3 rotation = Quaternion.Lerp(transform.rotation, lookRotation,
Time.deltaTime).eulerAngles;
        transform.rotation = Quaternion.Euler(rotation);
    }

    void rotateTarget()
    {
        if (Input.GetMouseButtonDown(0))
        {
            firstPos = Input.mousePosition;
        }
```

```
        if (Input.GetMouseButton(0))
        {
            secondPos = Input.mousePosition;
            deltaPos = (firstPos - secondPos).normalized;

            yForward += deltaPos.y;
            xRight += deltaPos.x;
        }

        if (Input.GetKey("w"))
        {
            yForward += 1f;
        }

        if (Input.GetKey("s"))
        {
            yForward -= 1f;
        }

        if (Input.GetKey("d"))
        {
            xRight += 1f;
        }

        if (Input.GetKey("a"))
        {
            xRight -= 1f;
        }

        target.rotation = Quaternion.Euler(yForward, xRight, 0f);
    }
}
```

Script yang digunakan untuk menggerakan Gambar 3D dilengkapi dengan fleksibel rotate dan zoom in and zoom out

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ScrollObject : MonoBehaviour
{
    Camera cam;

    public Transform target;

    Vector3 firstPos;
    Vector3 secondPos;

    Vector3 deltaPos;

    public float zoomSpeed;
    public float sensity = 5f;

    void Start()
    {
        cam = GetComponent<Camera>();
```

```
    }

     void Update()
    {
        AndroidController();
        AndroidZoom();
        MoveCamera();
    }

    void AndroidZoom()
    {
        if (Input.touchCount == 2)
        {
            Touch touch0 = Input.GetTouch(0);
            Touch touch1 = Input.GetTouch(1);

            Vector2 touch0_PrevPos = touch0.position - touch0.deltaPosition;
            Vector2 touch1_PrevPos = touch1.position - touch1.deltaPosition;

            float prevTouch_DeltaMag = (touch0_PrevPos -
touch1_PrevPos).magnitude;
            float touch_DeltaMag = (touch0.position -
touch1.position).magnitude;

            float deltaMag_Diff = prevTouch_DeltaMag - touch_DeltaMag;

            cam.fieldOfView += deltaMag_Diff * zoomSpeed;
            cam.fieldOfView = Mathf.Clamp(cam.fieldOfView, .1f, 179.9f);
        }
    }

    void AndroidController()
    {
        if (Input.touchCount > 0)
        {
            if (Input.GetTouch(0).phase == TouchPhase.Began)
            {
                firstPos = Input.GetTouch(0).position;
            }

            if (Input.GetTouch(0).phase == TouchPhase.Moved)
            {
                secondPos = Input.GetTouch(0).position;
                deltaPos = (firstPos - secondPos).normalized;

                transform.Translate(deltaPos * sensity * Time.deltaTime,
Space.Self);
                transform.LookAt(target);
            }
        }
    }

    void MoveCamera()
    {
        if (Input.GetMouseButtonDown(0))
        {
            firstPos = Input.mousePosition;
        }

        if (Input.GetMouseButton(0))
```

```
        {
            secondPos = Input.mousePosition;
            deltaPos = (firstPos - secondPos).normalized;

            transform.Translate(deltaPos * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("w"))
        {
            transform.Translate(Vector3.up * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("s"))
        {
            transform.Translate(Vector3.down * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("d"))
        {
            transform.Translate(Vector3.right * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("a"))
        {
            transform.Translate(Vector3.left * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("q"))
        {
            transform.Translate(Vector3.forward * sensity * Time.deltaTime,
Space.Self);
        }

        if (Input.GetKey("e"))
        {
            transform.Translate(Vector3.back * sensity * Time.deltaTime,
Space.Self);
        }

        transform.LookAt(target);
    }

}
```

Script ini digunakan pada scene Video yang bersifat Streaming Video

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.Video;
```

```
public class StreamVideo : MonoBehaviour {

    public RawImage rawImg;

    public VideoClip videoToPlay;

    private VideoPlayer videoPlayer;
    private VideoSource videoSource;

    private AudioSource audioSource;

    void Start()
    {
        Application.runInBackground = true;
        StartCoroutine(playVideo());
    }

    IEnumerator playVideo()
    {
        //Add VideoPlayer to the GameObject
        videoPlayer = gameObject.AddComponent<VideoPlayer>();

        //Add AudioSource
        audioSource = gameObject.AddComponent<AudioSource>();

        //Disable Play on Awake for both Video and Audio
        videoPlayer.playOnAwake = false;
        audioSource.playOnAwake = false;
        audioSource.Pause();

        //We want to play from video clip not from url

        videoPlayer.source = VideoSource.VideoClip;

        // Vide clip from Url
        //videoPlayer.source = VideoSource.Url;
        //videoPlayer.url =
"http://www.quirksmode.org/html5/videos/big_buck_bunny.mp4";


        //Set Audio Output to AudioSource
        videoPlayer.audioOutputMode = VideoAudioOutputMode.AudioSource;

        //Assign the Audio from Video to AudioSource to be played
        videoPlayer.EnableAudioTrack(0, true);
        videoPlayer.SetTargetAudioSource(0, audioSource);

        //Set video To Play then prepare Audio to prevent Buffering
        videoPlayer.clip = videoToPlay;
        videoPlayer.Prepare();

        //Wait until video is prepared
        WaitForSeconds waitTime = new WaitForSeconds(1);
        while (!videoPlayer.isPrepared)
        {
            Debug.Log("Preparing Video");
            //Prepare/Wait for 5 sceonds only
            yield return waitTime;
            //Break out of the while loop after 5 seconds wait
            break;
```

```
        }

        Debug.Log("Done Preparing Video");

        //Assign the Texture from Video to RawImage to be displayed
        rawImg.texture = videoPlayer.texture;

        //Play Video
        videoPlayer.Play();

        //Play Sound
        audioSource.Play();

        Debug.Log("Playing Video");
        while (videoPlayer.isPlaying)
        {
            Debug.LogWarning("Video Time: " +
Mathf.FloorToInt((float)videoPlayer.time));
            yield return null;
        }
        Debug.Log("Done Playing Video");
    }
}
```

Script yang Membuat Video menampilkan full screen pada layar HandPhone

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class VideoTest : MonoBehaviour {

    public string videoName = "";

    void Start ()
    {
        PlayMyVideo();
    }

    public void PlayMyVideo()
    {
        Handheld.PlayFullScreenMovie(videoName, Color.black,
FullScreenMovieControlMode.CancelOnInput,
FullScreenMovieScalingMode.AspectFit);
    }
}
```