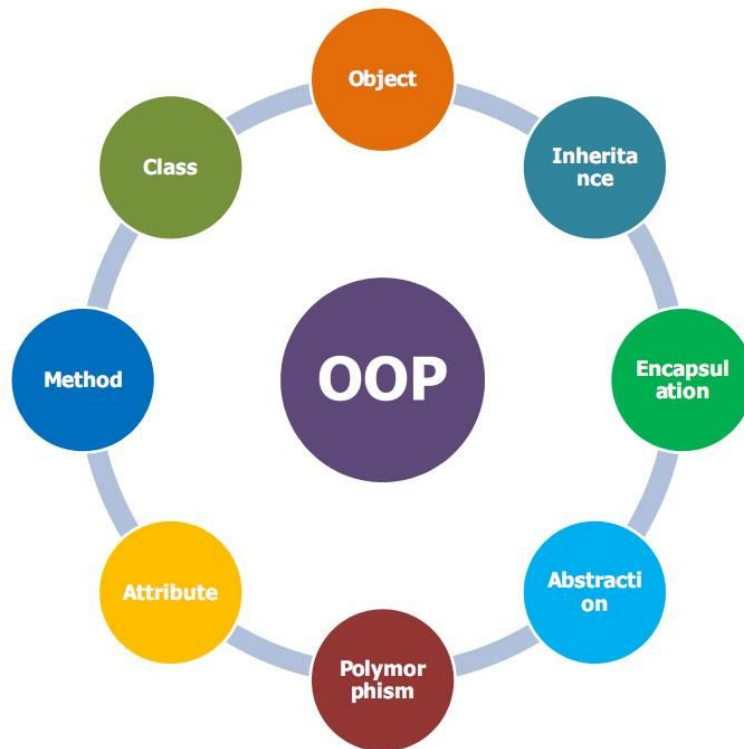


# OOP Øvelse 3 og ekstra Rapport

Lavet af Frederik Foss Nielsen



**C#** | Object  
Oriented  
Programming

# Inholdsfortegnelse

<b>Inholdsfortegnelse</b>	<b>2</b>
<b>Spørgsmål</b>	<b>3</b>
Hvad er fordelene med OOP?	3
Hvad er OOA?	3
Hvilke fordele har OOD?	3
Hvad er OOAD?	3
Forklar hvilke sammenhænge man bruger OOA og OOD?	3
Hvad kan man bruge OOP, OOA, OOD og OOAD?	3
Hvad er QA testning og hvad kan det bruges til?	3
<b>Opgaveformulering</b>	<b>3</b>
<b>Udførelse</b>	<b>3</b>
<b>Opgaveformulering</b>	<b>4</b>
<b>Udførelse</b>	<b>4</b>

# Øvelse 3

## Opgaveformulering

Du er nu klar til den afsluttende case øvelse, hvor der skal laves en lille C# programmer for virksomheden LAX A/S, som skal registrere de nomineringer film i en database og ved hjælp af et desktop programmer og en web version som kun visning de nomineringer film, LAX A/S har sat et krav om at det hele bliver skrevet i C# hvor der er bruger objektorienteret programmering som løsning, I skal selv komme med en plan, designe for programmet og en løsning forslag som skal fremlæggelse for instruktører før selve programmet programmeres og dokumenteres.

## Installationsvejledning

Alle ting på nær NodeJs og Unity var installeret, og at installere de to er så simpelt som at hente dem og så bare klikke på næste knappen indtil de er installeret.

## UML

Film		
ID	int	primary
Titel		varchar
Year		Year
Genre		Text
Skuespillere		Text

# Beskrivelse af objekter

## C# Program

Jeg har et objekt i mit c# program som hedder "Connection". Jeg bruger objektet til at opbevare forbindelses data såsom brugernavn og adgangskode. Derudover har jeg 4 metoder. den første metode "TestConn" bliver brugt når klassen bliver oprettet til at teste om brugernavnet og adgangskoden er korrekt indtastet. Reload er den næste metode og den sørger for at få alle data fra Film tabellen i mysql. Delete metoden bruges til at Slette en række i tabellen ud fra det id der er blevet angivet og så resetter den også Auto\_increment. Den sidste metode hedder insert og bruges til at sætte data ind i tabellen.

## Konklusion

Min konklusion er at det har været lidt kedeligt at lave den første del af øvelsen. Da det ikke er nogen ny ting mere at oprette forbindelse til en sql og så ændre nogle tabeller. Så måske hvis opgaven var omvendt kunne det have gjort den mere spændende. På den måde at det så ville have været Web appen som skulle kunne redigere data og c# programmet kun skulle kunne vise data. Men at lave Web appen var fedt, men også fordi det var lidt mere frit, da man selv kunne bestemme sprog så længe det havde med OOP at gøre. Jeg valgte at skrive Web appen i NodeJs som er Javascript som køre som server.

## Guide (Brugervejledning)

### C# Program

Der er billeder i bilag som viser de forskellige dele af programmet

Når programmet starter har du mulighed for at logge ind på serveren ved at skrive din Adgangskode og Brugernavn ovre til venstre af programmet. Derefter kan man klikke på Login knappen. Når man er logget ind har man mulighed for at Logge ud, Oprette, Slette og Genindlæse data. Hvis man klikker på Log Ud vil programmet logge ud og så kan man indtaste en anden bruger hvis man vil. Hvis man vil oprette data i tabellen kan man klikke på opret. Der vil så komme et nyt vindue frem hvor man så kan indtaste de forskellige værdier. Når det er gjort kan man klikke på opret for at tilføje det til sql. Hvis man i stedet vil slette kan man klikke på slet knappen, der vil så blive åbnet et nyt vindue hvor man så kan vælge et id og derefter slette det valgte ved at klikke på slet knappen. Hvis du bare vil tjekke om der er kommet nyt data i tabellen kan du klikke på genindlæs.

### Web APP

Web Appen har ingen funktionaliteter så kan ikke rigtig lave en guide til den.

# Øvelse Ekstra

## Opgaveformulering

Den afsluttende øvelse som er en ekstra øvelse som tager udgangspunkt i at bruge OOP teknikker i Unity hvor C# indgår, der skal laves en lille spil hvor du hopper med en bold eller element på skærmen, du skal selv komme med løsningen i denne øvelse.

## Unity

Unity er en spil engine som man kan bruge til at lave 3D og 2D spil. Det er lavet af Unity Technologies som er et danskejet firma. Unity kan lave spil til mange platforme såsom Windows, Mac Os, Xbox, Playstation og flere. Man kan programmere spillene i C# og det kan gøres i enten monodevelop eller i visual studio.

## Beskrivelse af objekter

I dette spil har jeg ikke selv lavet nogen objekter men i stedet brugt GameObject da jeg ikke havde brug for at have flere af den samme slags. GameObject er det objekt som alle ting i probs i unity engine har som standard og da mit program ikke skulle andet end at rykke spilleren og teste om den havde ramt noget. Var der ingen grund til at lave et objekt.

## Instantiering af et gameobject

```
GameObject bulletGo = (GameObject)Instantiate(bulletPrefab, firePoint.position,  
firePoint.rotation)  
Bullet bullet = bulletGo.GetComponent<Bullet>();
```

Her er et eksempel af hvordan man laver et nyt GameObject i Unity og henter komponenterne fra en klasse som hedder Bullet. Det der sker er at man laver et nyt objekt fysisk i spille og med det samme giver man en klasse som den skal bruge. I den klasse er der så en update som unity sørger for at køre hver frame som unity renderer.

# Kilder

## Øvelse 3

### C#

#### Startside.cs

```
using System;
using System.Windows.Forms;

namespace Nominering
{
    public partial class Startside : Form
    {
        public Startside()
        {
            InitializeComponent();
        }
        Connection Connect;
        //denne metode tjekker om koden og brugernavnet er korrekt og hvis ikke viser den en besked
        private void btnLogin_Click(object sender, EventArgs e)
        {
            Connect = new Connection(username.Text, password.Text, "192.168.0.37");
            if (Connect.ConnWorking)
            {
                btnCreate.Enabled = true;
                btnDelete.Enabled = true;
                btnLogin.Enabled = false;
                btnLogout.Enabled = true;
                btnReload.Enabled = true;
                username.Enabled = false;
                password.Enabled = false;
                password.Text = null;
                Grid.Enabled = true;
                Grid.DataSource = Connection.Reload();
            }
            else
            {
                MessageBox.Show(Connect.ex.Message);
            }
        }
        //denne metode disables knapperne der ikke er nødvendige og logger personen af sql serveren
        private void btnLogout_Click(object sender, EventArgs e)
        {
            btnCreate.Enabled = false;
            btnDelete.Enabled = false;
            btnLogin.Enabled = true;
            btnLogout.Enabled = false;
            btnReload.Enabled = false;
            username.Enabled = true;
        }
    }
}
```

```

        password.Enabled = true;
        Grid.Enabled = false;
        Grid.DataSource = null;
        Connect = null;
    }
    //denne metode reloader dataene fra sql serveren
    private void btnReload_Click(object sender, EventArgs e)
    {
        Grid.DataSource = Connection.Reload();
    }
    //denne metode laver et nyt vindue kaldet DeleteForm og reloader derefter data
    private void btnDelete_Click(object sender, EventArgs e)
    {
        new DeleteForm().ShowDialog();
        Grid.DataSource = Connection.Reload();
    }
    //denne metode laver et nyt vindue kaldet Insert og reloader derefter data
    private void btnCreate_Click(object sender, EventArgs e)
    {
        new Insert().ShowDialog();
        Grid.DataSource = Connection.Reload();
    }
}

```

## Insert.cs

```

using System;
using System.Collections.Generic;
using System.Windows.Forms;

namespace Nominering
{
    public partial class Insert : Form
    {
        public Insert()
        {
            InitializeComponent();
        }
        //denne metode tjekker at alle felter er udfyldt og derefter opretter den en ny række i
        //tabellen
        private void btnInsert_Click(object sender, EventArgs e)
        {
            if(Titel.Text.Length == 0 || Year.SelectedItem == null || Genre.Text.Length ==
0 || Skuespillere.Text.Length == 0)
            {
                MessageBox.Show("Udfyld venligst alle felter");
            }
            else
            {
                List<string> values = new List<string>();
                values.Add(Titel.Text);
                values.Add(Year.SelectedItem.ToString());
                values.Add(Genre.Text);
                values.Add(Skuespillere.Text);
                if (Connection.Insert(values) == null)
                {
                    this.Close();
                }
            }
        }
    }
}

```

```

    }
    else
    {
        MessageBox.Show("Der skete en fejl");
    }
}
}
}
}
}

```

## DeleteForm.cs

```

using System;
using System.Data;
using System.Windows.Forms;

namespace Nominering
{
    public partial class DeleteForm : Form
    {
        public DeleteForm()
        {
            InitializeComponent();
            GetID();
        }
        //denne metode sørger for at slette en række og samtidig checke om et id er valgt
        private void btnDelete_Click(object sender, EventArgs e)
        {
            if (IDBox.SelectedItem == null)
            {
                MessageBox.Show("Vælg venligst et ID");
            }
            else
            {
                int N;
                if(IDBox.SelectedIndex == IDBox.Items.Count - 1)
                {
                    N = Convert.ToInt32(IDBox.Items[IDBox.Items.Count - 1]);
                }
                else
                {
                    N = Convert.ToInt32(IDBox.Items[IDBox.Items.Count - 1]) + 1;
                }

                if(Connection.Delete(Convert.ToInt32(IDBox.SelectedItem.ToString()), N) ==
null)
                {
                    this.Close();
                }
                MessageBox.Show("Der er sket en fejl");
            }
        }
    }
}

```



```

    }
    //denne metode henter alle ID i film tabellen og sætter dem ind i en selector box
    private void GetID()
    {
        DataTable dt = Connection.Reload();
        foreach (DataRow row in dt.Rows)
            IDBox.Items.Add((int)row["ID"]);
    }
}

```

## Connection.cs

```

using System;
using System.Collections.Generic;
using System.Data;
using MySql.Data.MySqlClient;

namespace Nominering
{
    class Connection
    {
        static public MySqlConnection Conn { get; set; }
        public bool ConnWorking { get; set; }
        public Exception ex { get; set; }
        //denne constructor opretter klassen
        public Connection(string username, string password, string server)
        {
            string database = "Nominerede";
            string uid = username;
            string connectionString;
            connectionString = "SERVER=" + server + ";" + "DATABASE=" +
                database + ";" + "UID=" + uid + ";" + "PASSWORD=" + password + ";";
            Conn = new MySqlConnection(connectionString);
            TestConn();
        }
        private void TestConn()//denne metode tjekker om koden og brugernavnet er korrekt
        {
            try
            {
                Conn.Open();
                ConnWorking = true;
                ex = null;
            }
            catch(Exception e)
            {
                ex = e;
            }
            finally
            {
                Conn.Close();
            }
        }
        //denne metode henter data fra Film tabellen og returner dem
        static public DataTable Reload()
        {
            DataTable dt = new DataTable();

```

```
        try
        {
            using (MySqlDataAdapter da = new MySqlDataAdapter("SELECT * FROM Film",
Conn))
            {
                Conn.Open();
                using (dt)
                {
                    da.Fill(dt);
                }
            }
        }
        finally
        {
            Conn.Close();
        }
        return dt;
    }
}

// denne metode sletter en række i film udfra hvilket id der er angivet
static public Exception Delete(int ID, int N)
{
    MySqlCommand cmd = new MySqlCommand("DELETE FROM Film WHERE ID = " + ID
+ "; ALTER TABLE Film AUTO_INCREMENT = "+ N + ";", Conn );
    try
    {
        Conn.Open();
        cmd.ExecuteNonQuery();
    }
    catch (Exception e)
    {
        return e;
    }
    finally
    {
        Conn.Close();
    }
    return null;
}

// denne metode indsætter en række i tabellen med de data den får i listen
static public Exception Insert(List<string> values)
{
    MySqlCommand cmd = new MySqlCommand("INSERT INTO Film
(Titel,Year,Genre,Skuespillere)" +
$"VALUES ('{values[0]}','{values[1]}','{values[2]}','{values[3]}')", Conn);
    try
    {
        Conn.Open();
        cmd.ExecuteNonQuery();
    }
    catch (Exception e)
    {
        return e;
    }
    finally
    {
    }
```

```

        {
            Conn.Close();
        }
        return null;
    }
}
}

```

## Node JS

### Style.css

```

#nominerede {
    font-family: "Trebuchet MS", Arial, Helvetica, sans-serif;
    border-collapse: collapse;
    width: 100%;
}

#nominerede td, #nominerede th {
    border: 1px solid #ddd;
    padding: 8px;
}

#nominerede tr:nth-child(even){background-color: #f2f2f2;}

#nominerede tr:hover {background-color: #ddd;}

#nominerede th {
    padding-top: 12px;
    padding-bottom: 12px;
    text-align: left;
    background-color: #4CAF50;
    color: white;
}

```

### Index.ejs

```

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Nomineringer</title>
    <link rel="stylesheet" href="/style.css">
  </head>
  <body>
    <h1>Nomineringer</h1>
    <table style="width:100%;" id="nominerede">
      <th>ID</th>
      <th>Titel</th>
      <th>Year</th>
      <th>Genre</th>
      <th>Skuespillere</th>
      <%for (var i = 0; i < result.length; i++) { %>
      <tr>

```

```

        <td><%=result[i]["ID"]%></td>
        <td><%=result[i]["Titel"]%></td>
        <td><%=result[i]["Year"]%></td>
        <td><%=result[i]["Genre"]%></td>
        <td><%=result[i]["Skuespillere"]%></td>
    </tr>
    <% } %>
</table>
</body>
</html>

```

## App.js

```

var bodyParser    = require("body-parser"),
    express       = require("express"),
    mysql         = require('mysql'),
    app           = express();

app.set("view engine", "ejs"); //denne her function gør så view engine er ejs
app.set("views", __dirname + "/views"); //sætter html filens standard mappe
app.use(express.static(__dirname + "/public")); //sætter alle link og src filers standard
destination

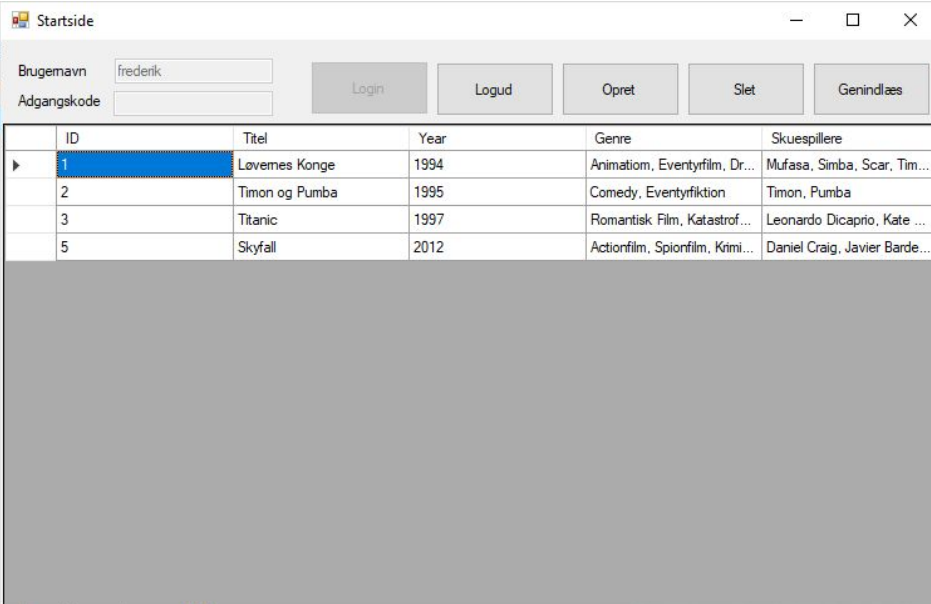
app.get("/", function (req, res) { //Denne her function henter dataene fra databasen når en
bruger prøver at komme ind på siden og så sender dataene videre til brugeren
    var con = mysql.createConnection({ //opretter et object som bruges til at oprette
forbindelse til serveren
        host: "192.168.0.37",
        user: "frederik",
        password: "Passwørd"
    });
    con.query("SELECT * FROM Nominerede.Film;", function (err, result) {
        con.end();
        if(err){
            res.send("Der er sket en fejl, prøv igen senere");
            return;
        }
        res.render("index",{result: result});
    });
});

app.listen(80, "192.168.0.143", function () { //Denne her function er den der gør at man
kan forbinde til webserveren
    console.log("Server has started!!!");
});

```

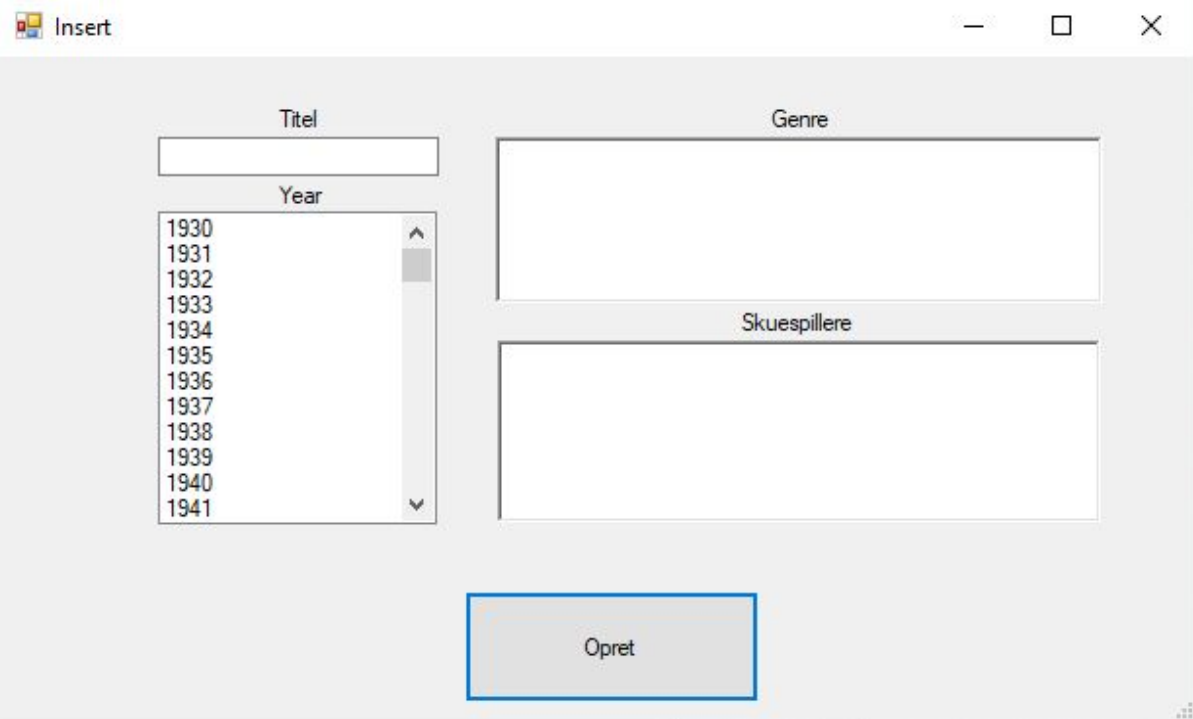
## Billeder

## Startside



ID	Titel	Year	Genre	Skuespillere
1	Lovemes Konge	1994	Animation, Eventyrfilm, Dr...	Mufasa, Simba, Scar, Tim...
2	Timon og Pumba	1995	Comedy, Eventyrfiktion	Timon, Pumba
3	Titanic	1997	Romantisk Film, Katastrof...	Leonardo Dicaprio, Kate ...
5	Skyfall	2012	Actionfilm, Spionfilm, Krimi...	Daniel Craig, Javier Barde...

## Insert



Titel

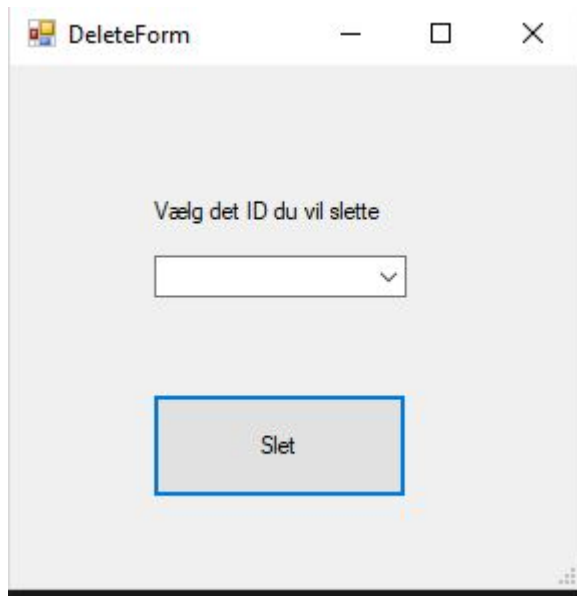
Year

Genre

Skuespillere

Opret

## Delete



## Øvelse Ekstra

## Scripts

## GameManager.cs

```
using UnityEngine;
using UnityEngine.SceneManagement;

public class GameManager : MonoBehaviour {

    bool GameHasEnded = false;

    public float RestartDelay = 1f;
    //Brugt til kalde Restart metoden når spillet er tabt
    public void EndGame ()
    {
        if (GameHasEnded == false)
        {
            GameHasEnded = true;
            Debug.Log("Game over!!");
            Invoke("Restart", RestartDelay);
        }
    }
    //Brugt til at genstarte spil
    void Restart ()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name);
    }
}
```

## PlayerCollision.cs

```
using UnityEngine;

public class PlayerCollision : MonoBehaviour {

    public PlayerMovement movement;

    //tjekker om bolden har ramt noget
    void OnCollisionEnter (Collision CollisionInfo)
    {
        if (CollisionInfo.collider.tag == "Obstacle")
        {
            movement.enabled = false;
            FindObjectOfType<GameManager>().EndGame(); ;
        }
    }
}
```

## PlayerMovement.cs

```
using UnityEngine;

public class PlayerMovement : MonoBehaviour {

    public Rigidbody rb;

    public float ForwardForce = 2000f;
    public float sidewaysForce = 500f;

    //Denne metode gør så spilleren rykker sig frem, ventsre og højre ud fra om a eller d
    er presset
    void FixedUpdate ()
    {
        rb.AddForce(0, 0, ForwardForce * Time.deltaTime, ForceMode.Force);

        if (Input.GetKey("d"))
        {
            rb.AddForce(sidewaysForce * Time.deltaTime, 0, 0, ForceMode.VelocityChange);
        }
        if (Input.GetKey("a"))
        {
            rb.AddForce(-sidewaysForce * Time.deltaTime, 0, 0, ForceMode.VelocityChange);
        }
    }
}
```

## Score.cs

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

```
using UnityEngine.UI;

public class Score : MonoBehaviour {

    public Transform player;
    public Text scoreText;
    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {
        scoreText.text = player.position.z.ToString("0");
    }
}
```

### FollowPlayer.cs

```
using UnityEngine;

public class FollowPlayer : MonoBehaviour {

    public Transform Player;
    public Vector3 offset;
    // Update is called once per frame
    void Update () {

        transform.position = Player.position + offset;
    }
}
```

### Billeder

