MED2 - Programming of mobile applications exam

- Frederik Ljungquist, 20212691 -

Code Improvements from Code Jam mini-project created by

Group 208

Aalborg University
Medialogy, 2. semester

Contents

1 Technical Information of the application Digital Study Card

1.1 As seen in ReadMe file

This is the Link to the Github reposotory:
https://github.com/Jung21/Code-Exam—Frederik-Ljungquist.git

This is the digital version of the study card at Aalborg University Copenhagen (AAU CPH), originally made by group 208 and modified by Frederik Ljungquist.
An APK called "Code-Exam—Frederik-Ljungquist, 0212691, Executable APK" is found as part of the uploaded material.
To build the application, an android phone must be connected. In Player Settings -> Publishing Settings, the keystore code is "123456" and project code is "123456".

In order to press the play button without errors and run the application in Unity, all gyroscope elements must be disabled and 3 packages are required. These are to be found in package manager as:

-Vector Graphics In order to see the svg. files in Unity add (com.unity.vectorgraphics)

-Input Manager The new input system that is used for the application (found in the Unity package repository)

-Device Simulator Used to simulate a mobile phone as the game view (found in the Unity package repository)

The relevant scripts and scene for the exam of Frederik Ljungquist must be found like this "Code-Exam—Frederik-Ljungquist"->"Assets"->"Scripts and scene relevant for exam". The script are attached to three Button gameobjects in the scene "RumOversigt" as seen in Figure 1

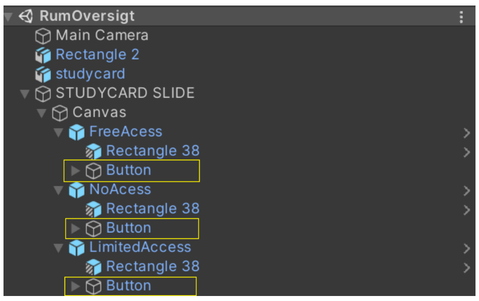
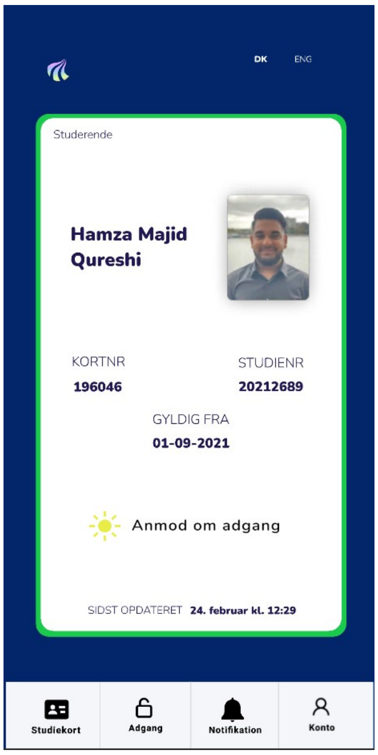


Figure 1: The three buttons where the three coresponding scripts are attached

1.2 Purpose of original software

The purpose of the the software is to create a digital version of the Study card at AAU CPH. The application uses gyroscope data, to create a colorchanging border around the "Card" as a means of authentication as seen in Figure 2 of the front page of the application. Furthermore the application uses the new input system to respond to touch.


DK ENG

Studerende

Hamza Majid Qureshi

KORTNR 196046

STUDIENR 20212689

GYLDIG FRA 01-09-2021

Anmod om adgang

SIDST OPDATERET **24. februar kl. 12:29**

Figure 2: This Figure shows the frontpage of the Study Card Application

The Studycard shows a picture of the student, their Card Number, their Study Number and when the Card was made valid. The menu bar at the button would use touch controls to go to different scenes, though "Konto" is only a picture.
One of the key features of the application was to be able to mimic the interaction with NFC scan able objects, this was acheived by touching the study card where a message with access information would be spawned as a pop-up notification.

By touching the study card a message will appear with access information. One of three types of messages can be chosen from the inspector, these being one of the following three:

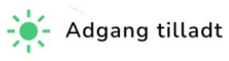


Figure 3: The message indicating free access

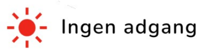


Figure 4: The message indicating no access

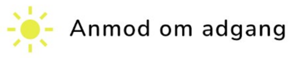
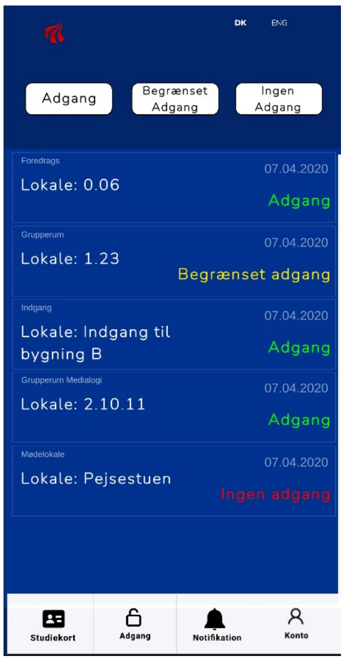


Figure 5: The message indicating limited access

For each type of message appearing a specific sound would play, as a further way to indicate the type of access. Furthermore a scene with information about rooms and their access level was created as seen in Figure 6, as a means of the user knowing which doors they could access, have limited access to or could not access . This was the feature that I worked on during Code Jam, and that I chose to improve for this Exam.


DK ENG

Adgang | Begrænset Adgang | Ingen Adgang

Foredrags
Lokale: 0.06
07.04.2020
Adgang

Grupperum
Lokale: 1.23
07.04.2020
Begrænset adgang

Indgang
Lokale: Indgang til bygning B
07.04.2020
Adgang

Grupperum Medialogi
Lokale: 2.10.11
07.04.2020
Adgang

Mødelokale
Lokale: Pejsestuen
07.04.2020
Ingen adgang

Studiekort | Adgang | Notifikation | Konto

Figure 6: This Figure shows the "Adgang" tab of the Study Card Application

1.3 Relevant UML Diagrams

The UML Case diagram as seen as Figure 7 shows the interaction between the user and the scripts in the Study Card Application.

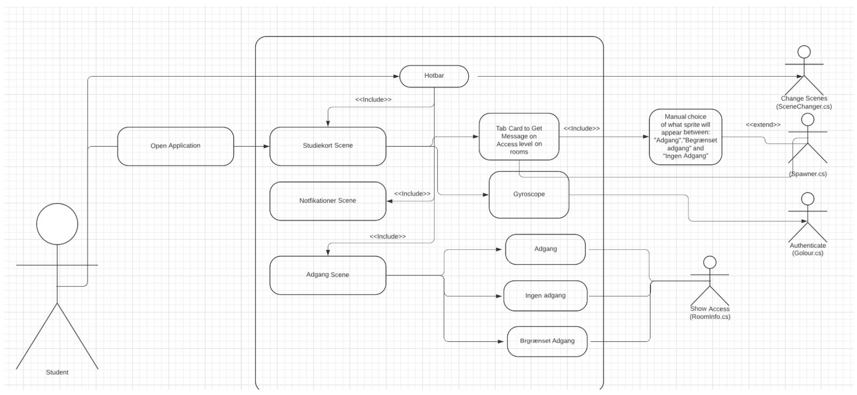


Figure 7: This Figure shows the UML Case diagram of the Study Card Application

Figure 8 shows a Class diagram detailing the relationship between the different scripts in the application. It is to be noted that for this paper that the RoomInfo script has been changed to be abstract with three classes inheriting from it. This was changed in the Class diagram according to the changes made by Frederik Ljungquist.

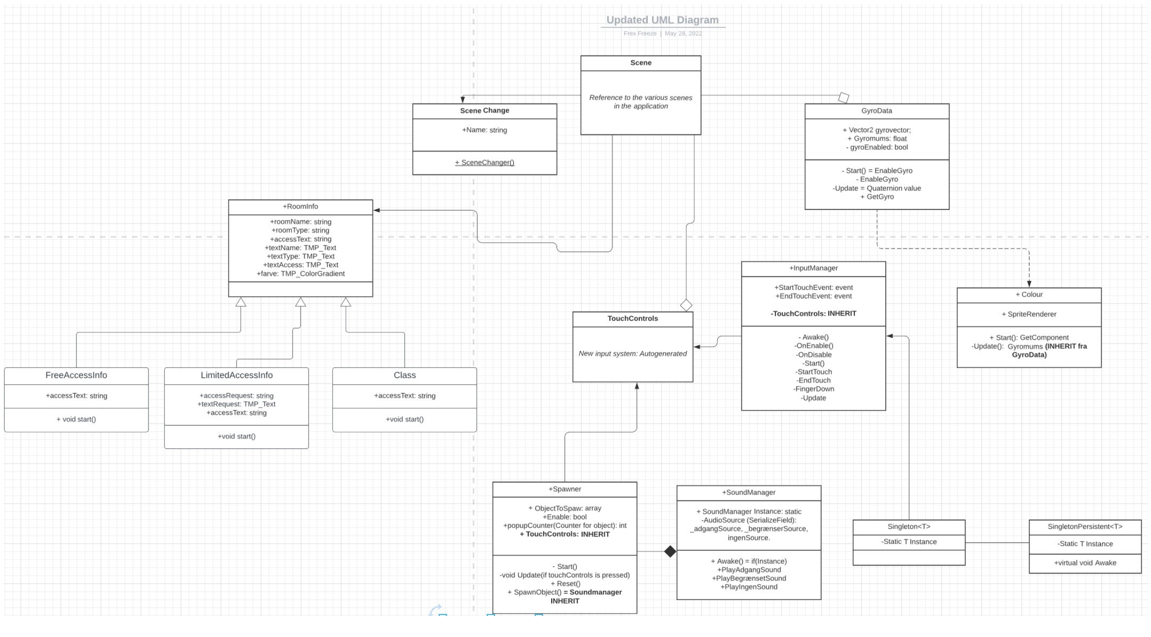
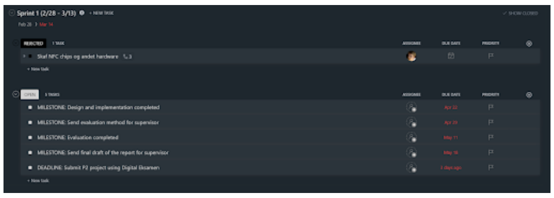


Figure 8: This Figure shows the UML Class diagram of the Study Card Application

1.4 Scrum Sprint planning

Figures 9, 10, 11, 12, shows the Scrum Sprint plans made with the website ClickUp (https://clickup.com). These plans was not made as a calendar, but was instead set up as tasks needed to be completed in a two week sprint and tasks had points assigned to them in order to judge how much time it would be required to complete them. This worked for the first two sprints though as it can be seen, fewer tasks was set and ultimately none was made. This in part happened due to the unpredictability of the project. Many tasks would appear during a sprint which could not be planned for, thus derailing the time to complete all tasks. Scrum was ultimately dropped though a calendar for the last month of the project was created as seen in Figure 13, this was not followed and the project was completed without the use the plan.

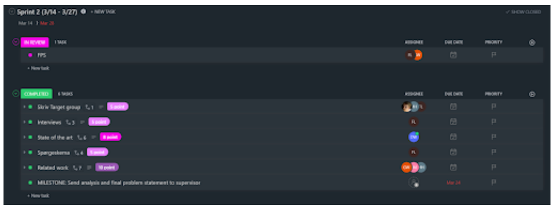


Sprint 1 (2/28 - 3/13) + NEW TASK ✓ SHOW CLOSED
Feb 28 → Mar 14

REJECTED 1 TASK	ASSIGNEE	DUE DATE	PRIORITY
Skaf NFC chips og andet hardware 3			
+ New task			

OPEN 5 TASKS	ASSIGNEE	DUE DATE	PRIORITY
MILESTONE: Design and implementation completed		Apr 22	
MILESTONE: Send evaluation method for supervisor		Apr 29	
MILESTONE: Evaluation completed		May 11	
MILESTONE: Send final draft of the report for supervisor		May 18	
DEADLINE: Submit P2 project using Digital Eksamen		3 days ago	
+ New task			

Figure 9: This Figure shows the first Scrum Sprint plan. Made in ClickUp

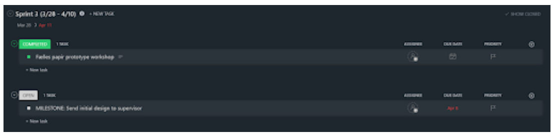


Sprint 2 (3/14 - 3/27) + NEW TASK ✓ SHOW CLOSED
Mar 14 → Mar 28

IN REVIEW 1 TASK	ASSIGNEE	DUE DATE	PRIORITY
FPS	FL GW		
+ New task			

COMPLETED 6 TASKS	ASSIGNEE	DUE DATE	PRIORITY
Skriv Target group 1 5 point			
Interviews 3 5 point	FL		
State of the art 6 8 point	DW		
Spørgeskema 4 5 point	FL		
Related work 7 10 point	GW		
MILESTONE: Send analysis and final problem statement to supervisor		Mar 24	
+ New task			

Figure 10: This Figure shows the second Scrum Sprint plan. Made in ClickUp

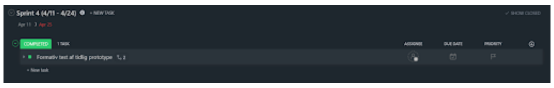


Sprint 3 (3/28 - 4/10) + NEW TASK ✓ SHOW CLOSED
Mar 28 → Apr 11

COMPLETED 1 TASK	ASSIGNEE	DUE DATE	PRIORITY
Fælles papir prototype workshop			
+ New task			

OPEN 1 TASK	ASSIGNEE	DUE DATE	PRIORITY
MILESTONE: Send initial design to supervisor		Apr 8	
+ New task			

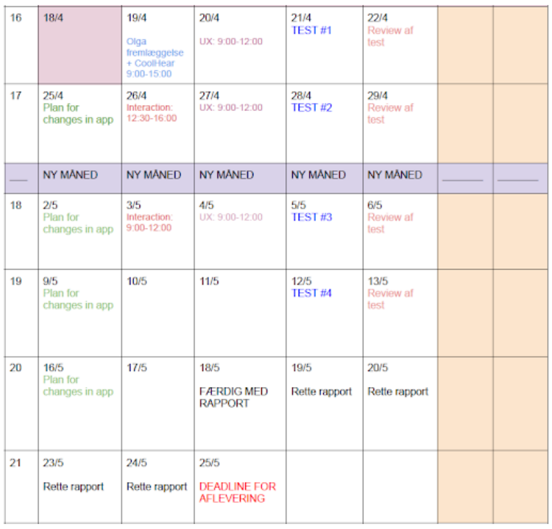
Figure 11: This Figure shows the third Scrum Sprint plan. Made in ClickUp



Sprint 4 (4/11 - 4/24) + NEW TASK ✓ SHOW CLOSED
Apr 11 → Apr 25

COMPLETED 1 TASK	ASSIGNEE	DUE DATE	PRIORITY
Formativ test af tidlig prototype 2			
+ New task			

Figure 12: This Figure shows the fourth Scrum Sprint plan. Made in Clickup

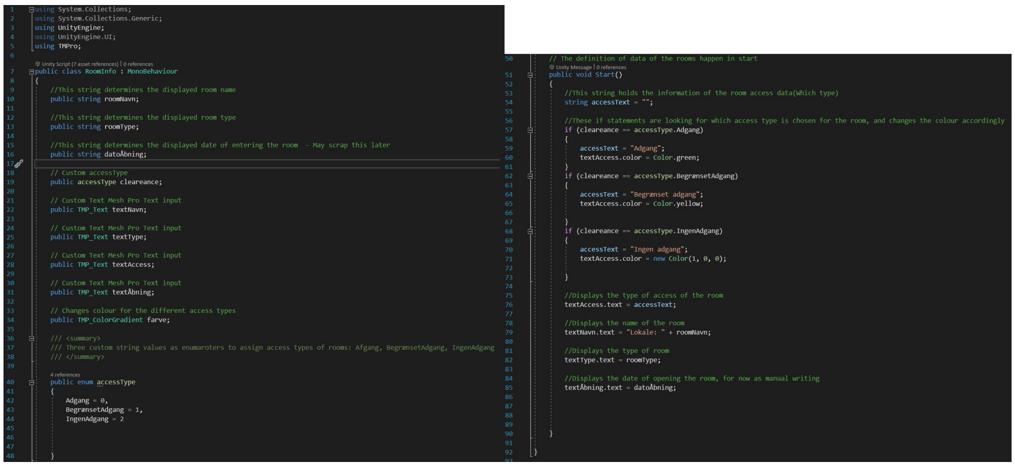


16	18/4	19/4 Olga fremlæggelse + CoolHear 9:00-15:00	20/4 UX: 9:00-12:00	21/4 TEST #1	22/4 Review af test		
17	25/4 Plan for changes in app	26/4 Interaction: 12:30-16:00	27/4 UX: 9:00-12:00	28/4 TEST #2	29/4 Review af test		
___	NY MÅNED	NY MÅNED	NY MÅNED	NY MÅNED	NY MÅNED	______	______
18	2/5 Plan for changes in app	3/5 Interaction: 9:00-12:00	4/5 UX: 9:00-12:00	5/5 TEST #3	6/5 Review af test		
19	9/5 Plan for changes in app	10/5	11/5	12/5 TEST #4	13/5 Review af test		
20	16/5 Plan for changes in app	17/5	18/5 FÆRDIG MED RAPPORT	19/5 Rette rapport	20/5 Rette rapport		
21	23/5 Rette rapport	24/5 Rette rapport	25/5 DEADLINE FOR AFLEVERING				

Figure 13: This Figure shows the fifth Scrum Sprint plan. Made in Excel

1.5 Discussion of Code Improvement by Frederik Ljungquist

The code I created during was the script RoomInfo as seen in Figure 14. This code used an Enum to determine three different types of access that three if statements look for. Corresponding to the enum three different types of access will be written and coloured as TextMeshPro as seen in Figure 6.
This was a simple script which worked as intended though some parts could be better optimized, one example being the string seen on Line 54 of Figure 14. This string was unnecessary as seen on Line 28 of Figure 14 a TMP Text variable, testAccess, was already created.

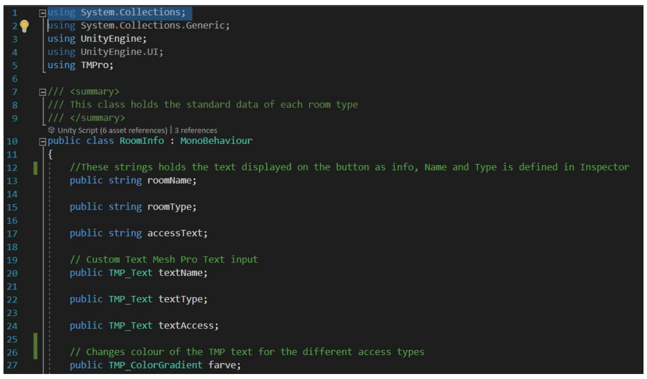


```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.UI;
5  using TMPro;
6
   Unity Script (7 asset references) | 0 references
7  public class RoomInfo : MonoBehaviour
8  {
9      //This string determines the displayed room name
10     public string roomNavn;
11
12     //This string determines the displayed room type
13     public string roomType;
14
15     //This string determines the displayed date of entering the room  - May scrap this later
16     public string datoÅbning;
17
18     // Custom accessType
19     public accessType cleareance;
20
21     // Custom Text Mesh Pro Text input
22     public TMP_Text textNavn;
23
24     // Custom Text Mesh Pro Text input
25     public TMP_Text textType;
26
27     // Custom Text Mesh Pro Text input
28     public TMP_Text textAccess;
29
30     // Custom Text Mesh Pro Text input
31     public TMP_Text textÅbning;
32
33     // Changes colour for the different access types
34     public TMP_ColorGradient farve;
35
36     /// <summary>
37     /// Three custom string values as enumaroters to assign access types of rooms: Afgang, BegrænsetAdgang, IngenAdgang
38     /// </summary>
39
       4 references
40     public enum accessType
41     {
42         Adgang = 0,
43         BegrænsetAdgang = 1,
44         IngenAdgang = 2
45
46
47
48     }
```

```
50     // The definition of data of the rooms happen in start
       Unity Message | 0 references
51     public void Start()
52     {
53         //This string holds the information of the room access data(Which type)
54         string accessText = "";
55
56         //These if statements are looking for which access type is chosen for the room, and changes the colour accordingly
57         if (cleareance == accessType.Adgang)
58         {
59             accessText = "Adgang";
60             textAccess.color = Color.green;
61         }
62         if (cleareance == accessType.BegrænsetAdgang)
63         {
64             accessText = "Begrænset adgang";
65             textAccess.color = Color.yellow;
66
67         }
68         if (cleareance == accessType.IngenAdgang)
69         {
70             accessText = "Ingen adgang";
71             textAccess.color = new Color(1, 0, 0);
72
73         }
74
75         //Displays the type of access of the room
76         textAccess.text = accessText;
77
78         //Displays the name of the room
79         textNavn.text = "Lokale: " + roomNavn;
80
81         //Displays the type of room
82         textType.text = roomType;
83
84         //Displays the date of opening the room, for now as manual writing
85         textÅbning.text = datoÅbning;
86
87
88
89
90     }
91
92 }
```

Figure 14: This Figure shows the orginal RoomInfo script

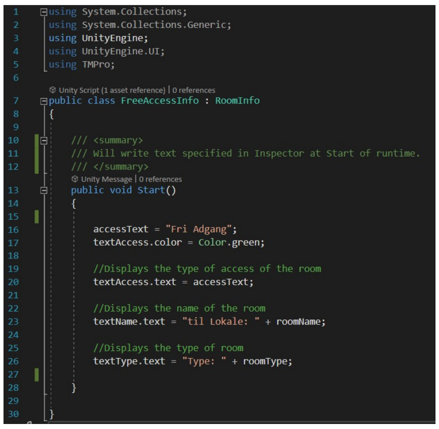
I chose to improve this script by changing RoomInfo to an abstract class with three new scripts inheriting from it as seen in Figure 15. This was ultimately done to increase flexibility of the code and make it possible for future functionality be specific to the different access types.



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

/// <summary>
/// This class holds the standard data of each room type
/// </summary>
Unity Script (6 asset references) | 3 references
public class RoomInfo : MonoBehaviour
{
    //These strings holds the text displayed on the button as info, Name and Type is defined in Inspector
    public string roomName;

    public string roomType;

    public string accessText;

    // Custom Text Mesh Pro Text input
    public TMP_Text textName;

    public TMP_Text textType;

    public TMP_Text textAccess;

    // Changes colour of the TMP text for the different access types
    public TMP_ColorGradient farve;
```

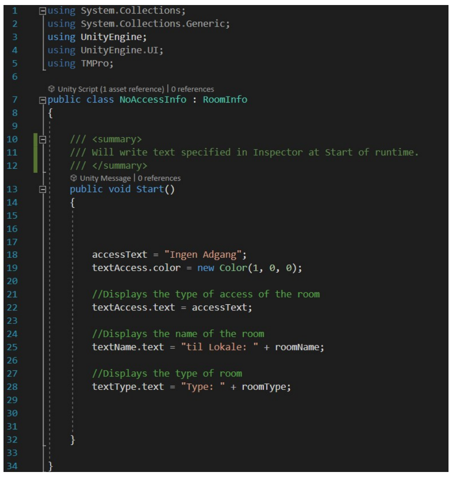
Figure 15: This Figure shows the new RoomInfo script. Which now only contains the shared variables of all Room Access types

Figure 16, 17 and 18 shows the three new scripts inheriting variables from the abstract class RoomInfo. As seen in Figure 18 this script contains unique variables from the other two scripts. This is given as an example of why the use of an abstract class will benefit the Information writing, as the three classes may need specific functionalities that the other two wont use.
The visual layout of the scene was also changed in Unity. As seen in Figure 19.



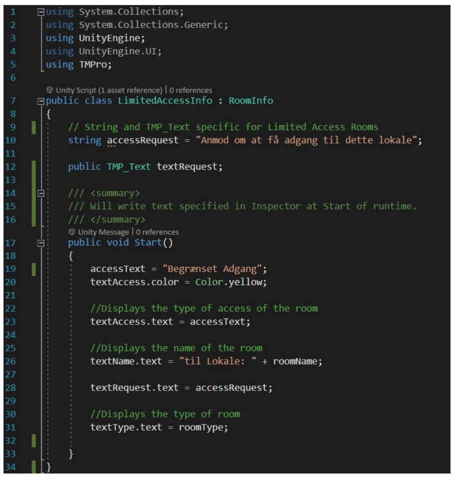
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;

Unity Script (1 asset reference) | 0 references
public class FreeAccessInfo : RoomInfo
{

    /// <summary>
    /// Will write text specified in Inspector at Start of runtime.
    /// </summary>
    Unity Message | 0 references
    public void Start()
    {

        accessText = "Fri Adgang";
        textAccess.color = Color.green;

        //Displays the type of access of the room
        textAccess.text = accessText;

        //Displays the name of the room
        textName.text = "til Lokale: " + roomName;

        //Displays the type of room
        textType.text = "Type: " + roomType;

    }

}
```

Figure 16: This Figure shows the script FreeAccessInfo. Which inherits variables from the abstract class RoomInfo and now contains the start defining the text written in the scene RumOverSigt



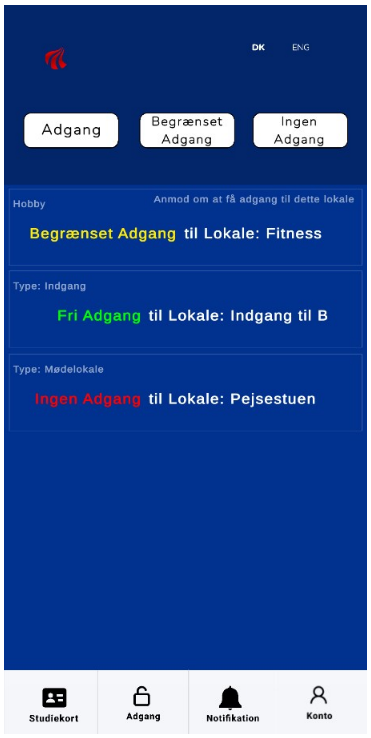
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;

Unity Script (1 asset reference) | 0 references
public class NoAccessInfo : RoomInfo
{

    /// <summary>
    /// Will write text specified in Inspector at Start of runtime.
    /// </summary>
    Unity Message | 0 references
    public void Start()
    {



        accessText = "Ingen Adgang";
        textAccess.color = new Color(1, 0, 0);

        //Displays the type of access of the room
        textAccess.text = accessText;

        //Displays the name of the room
        textName.text = "til Lokale: " + roomName;

        //Displays the type of room
        textType.text = "Type: " + roomType;



    }

}
```

Figure 17: This Figure shows the script NoAccessInfo. Which inherits variables from the abstract class RoomInfo and now contains the start defining the text written in the scene RumOverSigt



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

Unity Script (1 asset reference) | 0 references
public class LimitedAccessInfo : RoomInfo
{
    // String and TMP_Text specific for Limited Access Rooms
    string accessRequest = "Anmod om at få adgang til dette lokale";

    public TMP_Text textRequest;

    /// <summary>
    /// Will write text specified in Inspector at Start of runtime.
    /// </summary>
    Unity Message | 0 references
    public void Start()
    {
        accessText = "Begrænset Adgang";
        textAccess.color = Color.yellow;

        //Displays the type of access of the room
        textAccess.text = accessText;

        //Displays the name of the room
        textName.text = "til Lokale: " + roomName;

        textRequest.text = accessRequest;

        //Displays the type of room
        textType.text = roomType;

    }
}
```

Figure 18: This Figure shows the script LimitedAccessInfo. Which inherits variables from the abstract class RoomInfo and now contains the start defining the text written in the scene RumOverSigt



Hobby
Anmod om at få adgang til dette lokale
Begrænset Adgang til Lokale: Fitness

Type: Indgang
Fri Adgang til Lokale: Indgang til B

Type: Mødelokale
Ingen Adgang til Lokale: Pejsestuen

Figure 19: This Figure shows the the updtaed scene called "RumOversigt"

List of Figures