Class CheckAdminPanel

Class for handling if the password was allready correctly entered, if yes then directly show the adminpanel when clicking at menu.

Inheritance

↳ System.Object
  ↳ CheckAdminPanel

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
public class CheckAdminPanel : MonoBehaviour
```

Fields

AdminOptionsPanel

Link to the AdminOptionsPanel object to work with it.

Declaration

```
public GameObject AdminOptionsPanel
```

Field Value

Type	Description
GameObject	

Menu

Link to the Menu object to work with it.

Declaration

```
public GameObject Menu
```

Field Value

Type	Description
GameObject	

PasswordHandling

Link to the PasswordHandling object to work with it.

Declaration

```
public GameObject PasswordHandling
```

Field Value

Type	Description
GameObject	

PasswordItems

Link to the PasswordItems object to work with it.

Declaration

```
public GameObject PasswordItems
```

Field Value

Type	Description
GameObject	

Methods

allreadycommitedpassword()

Method to check if the passwordvariable from other script is one. When it is one, do not show the screen for the password input again.

Declaration

```
public void allreadycommitedpassword()
```

Class NavigationController

Class for handling the navigation line to show the player the way to the target.

Inheritance

↳ System.Object
  ↳ NavigationController

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
public class NavigationController : MonoBehaviour
```

Fields

line

Line created to show the player the way to the target.

Declaration

```
public LineRenderer line
```

Field Value

Type	Description
LineRenderer	

navigationYOffset

Slider to adapt the height of the line.

Declaration

```
public Slider navigationYOffset
```

Field Value

Type	Description
Slider	

path

Path created to show player the way to the target

Declaration

```
public NavMeshPath path
```

Field Value

Type	Description
NavMeshPath	

Properties

TargetPosition

Position of the selected target.

Declaration

```
public Vector3 TargetPosition { get; set; }
```

Property Value

Type	Description
Vector3	

Methods

AddLineOffset()

Method used to create an array of positions to create the path.

Declaration

```
public Vector3[] AddLineOffset()
```

Returns

Type	Description
Vector3[]	Returns an array of position were the line to show the path is placed.

Start()

When game starts a new NavMeshPath gets creates and the display dimming get disabled.

Declaration

```
public void Start()
```

ToggleLineVisibility()

Method for the tooglelinevisability button to show or hide the line towards the target.

Declaration

```
public void ToggleLineVisibility()
```

Update()

Method used to constantly update the path towards the target while player is standing or moving.

Declaration

```
public void Update()
```

Class PasswordController

Class for handling the passwort protection for the admin panel.

Inheritance

↳ System.Object
  ↳ PasswordController

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
public class PasswordController : MonoBehaviour
```

Fields

adminoptionspanel

Declaration

```
public GameObject adminoptionspanel
```

Field Value

Type	Description
GameObject	

button

Declaration

```
public GameObject button
```

Field Value

Type	Description
GameObject	

delbutton

Declaration

```
public GameObject delbutton
```

Field Value

Type	Description

Type	Description
GameObject	

PasswordInput

Declaration

```
public InputField PasswordInput
```

Field Value

Type	Description
InputField	

Methods

CheckPasswordCondition()

Method to declase the password and handle the password protection. If password correct show delete target button and the adminpanel. If password is not correct display "wrong passwort" in the input field and do nothing.

Declaration

```
public void CheckPasswordCondition()
```

Class PulsingLine

Class to make sure the line, shown when targets gets selected to navigate to, is pulsing.

Inheritance

↳ System.Object
  ↳ PulsingLine

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
public class PulsingLine : MonoBehaviour
```

Fields

endColor

Second color to be shown.

Declaration

```
public Color32 endColor
```

Field Value

Type	Description
Color32	

endPos

/// Current position of the selected target.

Declaration

```
public Vector3 endPos
```

Field Value

Type	Description
Vector3	

lastColorChangeTime

Time it took for the last color change.

Declaration

```
public float lastColorChangeTime
```

Field Value

Type	Description
System.Single	

material

/// Material to transform the color.

Declaration

```
public Material material
```

Field Value

Type	Description
Material	

myPointsInLine

/// Array of points frim the line.

Declaration

```
public Vector3[] myPointsInLine
```

Field Value

Type	Description
Vector3[]	

r

Object were changes should be applied.

Declaration

```
public LineRenderer r
```

Field Value

Type	Description
LineRenderer	

startColor

First color to be shown.

Declaration

```
public Color32 startColor
```

Field Value

Type	Description
Color32	

startPos

Current position of the player.

Declaration

```
public Vector3 startPos
```

Field Value

Type	Description
Vector3	

Methods

changecolor(Single)

This method used the lerp function to change between two color in different speeds so it shows the player how far he/she is away from the target.

Declaration

```
public void changecolor(float Fadeduration)
```

Parameters

Type	Name	Description
System.Single	*Fadeduration*	Gets the Fadeduration to know how fast the line should change its color

Start()

When the Game is started this method gets executed. It gets the linerenderer and his material.

Declaration

```
public void Start()
```

Update()

Method used to continously update the frequency the line changes is color so it can signal how far the target is. Gets to position of the player and the target and calculates the distance. The higher the distance the lower the frequency the line changes its color.

Declaration

```
public void Update()
```

Class QrCodeRecenter

Class for handling all actions concerning the qrcode scanning process.

Inheritance

↳ System.Object
  ↳ QrCodeRecenter

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
public class QrCodeRecenter : MonoBehaviour
```

Fields

cameraImageTexture

SaveImage variable.

Declaration

```
public Texture2D cameraImageTexture
```

Field Value

Type	Description
Texture2D	

cameraManager

/// Unity cameramanager for reading the frames with the qr code.

Declaration

```
public ARCameraManager cameraManager
```

Field Value

Type	Description
ARCameraManager	

qrCodeScanningPanel

Panel to show the player the qrcodescanner is active.

Declaration

```
public GameObject qrCodeScanningPanel
```

Field Value

Type	Description
GameObject	

reader

Create a barcode reader instance.

Declaration

```
public IBarcodeReader reader
```

Field Value

Type	Description
IBarcodeReader	

scanningEnabled

Variable the save if panel is enabled or not

Declaration

```
public bool scanningEnabled
```

Field Value

Type	Description
System.Boolean	

session

/// Unity ARSession with the environment.

Declaration

```
public ARSession session
```

Field Value

Type	Description
ARSession	

sessionOrigin

/// Origin of the AR Session.

Declaration

```
public ARSessionOrigin sessionOrigin
```

Field Value

Type	Description
ARSessionOrigin	

startitem

Reference to the gameobject for getting the position data etc.

Declaration

```
public TargetFacade startitem
```

Field Value

Type	Description
TargetFacade (DocFx.TargetFacade.html)	

Methods

OnCameraFrameReceived(ARCameraFrameEventArgs)

Method for handling all qr scanner related task. Method mainly from the following link with small changes. https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@1.0/manual/cpu-camera-image.html (https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@1.0/manual/cpu-camera-image.html)

Declaration

```
public void OnCameraFrameReceived(ARCameraFrameEventArgs eventArgs)
```

Parameters

Type	Name	Description
ARCameraFrameEventArgs	*eventArgs*	Event received by camera.

OnDisable()

When GameObject with script is not active do not send the frameReceived by the cameramanager.

Declaration

```
public void OnDisable()
```

OnEnable()

When GameObject with script is active send the frameReceived by the cameramanager.

Declaration

```
public void OnEnable()
```

SetQrCodeRecenterTarget(String)

Method for button to active and deactive scanning for the qr code otherwise application would crash.

Declaration

```
public void SetQrCodeRecenterTarget(string targetText)
```

Parameters

Type	Name	Description
System.String	*targetText*	string variable delivered by the qr scanner method to check if the scanned text matches the start passphrase.

ToggleScanning()

Method for button to active and deactive scanning for the qr code otherwise application would crash.

Declaration

```
public void ToggleScanning()
```

Class SetUiText

Class for handling the text field in the line debug options panel.

Inheritance

↳ System.Object
  ↳ SetUiText

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
public class SetUiText : MonoBehaviour
```

Methods

OnSliderValueChanged(Single)

Facade with all different information of the target. For better identification a name gets added.

Declaration

```
public void OnSliderValueChanged(float numericValue)
```

Parameters

Type	Name	Description
System.Single	*numericValue*	

Class Target

Target class to store the data for each target provided by the json and make it accessable for all different types of classes in unity. Name and Position gets added to added.

Inheritance

↳ System.Object
  ↳ Target

Inherited Members

System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.ToString()

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
[Serializable]
public class Target
```

Fields

Name

Declaration

```
public string Name
```

Field Value

Type	Description
System.String	

Position

Declaration

```
public Vector3 Position
```

Field Value

Type	Description

Type	Description
Vector3	

Class TargetFacade

Facade with all different information of the target. For better identification a name gets added.

Inheritance

↳ System.Object
  ↳ TargetFacade

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
[Serializable]
public class TargetFacade : MonoBehaviour
```

Fields

Name

Declaration

```
public string Name
```

Field Value

Type	Description
System.String	

Class TargetHandler

Class for handling all actions concerning the targets in the environment.

Inheritance

↳ System.Object
  ↳ TargetHandler

Namespace: DocFx (DocFx.html)

Assembly: DocFx.dll

Syntax

```
public class TargetHandler : MonoBehaviour
```

Fields

currentTargetItems

List to store current targets.

Declaration

```
public List<TargetFacade> currentTargetItems
```

Field Value

Type	Description
System.Collections.Generic.List<TargetFacade (DocFx.TargetFacade.html)>	

item

Variable to store the current selected target.

Declaration

```
public string item
```

Field Value

Type	Description
System.String	

LoadURL

Address for loading the data to create target gameobjects.

Declaration

```
public const string LoadURL = "https://ar-app-rest-api.herokuapp.com/positions"
```

Field Value

Type	Description
System.String	

navigationController

Instance were the class is initialized.

Declaration

```
public NavigationController navigationController
```

Field Value

Type	Description
NavigationController (DocFx.NavigationController.html)	

navigationXOffset

Slider to change x location of selected target.

Declaration

```
public Slider navigationXOffset
```

Field Value

Type	Description
Slider	

navigationZOffset

Slider to change z location of selected target.

Declaration

```
public Slider navigationZOffset
```

Field Value

Type	Description
Slider	

NewTargetNameInput

Input field to give the new target added a name.

Declaration

```
public InputField NewTargetNameInput
```

Field Value

Type	Description
InputField	

outputstring

String for the JSON ulpoaded to server when save gets clicked.

Declaration

```
public string outputstring
```

Field Value

Type	Description
System.String	

saveprevious

Save previous item to check if a new item was selected.

Declaration

```
public string saveprevious
```

Field Value

Type	Description
System.String	

SaveURL

Address for uploading the data when targets get updated in position etc..

Declaration

```
public const string SaveURL = "https://ar-app-rest-api.herokuapp.com/updateposition
s"
```

Field Value

Type	Description
System.String	

selected

Show if a new item gets selected by user to show location by changing color.

Declaration

```
public int selected
```

Field Value

Type	Description
System.Int32	

targetDataDropdown1

Dropdown menu filled with targest on first menu page.

Declaration

```
public TMP_Dropdown targetDataDropdown1
```

Field Value

Type	Description
TMP_Dropdown	

targetDataDropdown2

Dropdown menu filled with targets on last menu page.

Declaration

```
public TMP_Dropdown targetDataDropdown2
```

Field Value

Type	Description
TMP_Dropdown	

targetDataDropdown3

Dropdown menu filled with targets on adminpanel

Declaration

```
public TMP_Dropdown targetDataDropdown3
```

Field Value

Type	Description
TMP_Dropdown	

targetlist

List to store current targets.

Declaration

```
public List<Target> targetlist
```

Field Value

Type	Description
System.Collections.Generic.List<Target (DocFx.Target.html)>	

targetObjectPrefab

Prefab to create targets from.

Declaration

```
public GameObject targetObjectPrefab
```

Field Value

Type	Description
GameObject	

targetObjectsParentTransforms

Location were to store targets in environment.

Declaration

```
public Transform[] targetObjectsParentTransforms
```

Field Value

Type	Description
Transform[]	

targets

Array to store current targets.

Declaration

```
public Target[] targets
```

Field Value

Type	Description
Target (DocFx.Target.html)[]	

uniblue

Color of the blue used by the university.

Declaration

```
public Color32 uniblue
```

Field Value

Type	Description
Color32	

unired

Color of the red used by the university.

Declaration

```
public Color32 unired
```

Field Value

Type	Description
Color32	

value

Variable to store the selected value of a dropdown menu.

Declaration

```
public int value
```

Field Value

Type	Description
System.Int32	

Methods

addTarget()

Add a new target the the environment and update all dropdown menus.

Declaration

```
public void addTarget()
```

asyncGetRequest()

Method for getting the JSON Data from our WebServer.

Declaration

```
public async Task<string> asyncGetRequest()
```

Returns

Type	Description
System.Threading.Tasks.Task<System.String>	Returns the JSON as a Task string.

CreateTargetFacade(Target)

Creates the target in the environment with all given data from the parameter.

Declaration

```
public TargetFacade CreateTargetFacade(Target target)
```

Parameters

Type	Name	Description
Target (DocFx.Target.html)	*target*	Gets a target which gets created during method.

Returns

Type	Description
TargetFacade (DocFx.TargetFacade.html)	TargetFacade of each creates target.

FillDropdownWithTargetItems()

Fills both dropdowns with the names of the provided targets.

Declaration

```
public void FillDropdownWithTargetItems()
```

GenerateTargetItems()

Load the targets from the json to an array of targets and create targets in environment. Set the sliders to change the coordinates of the target selected. Update the Dropdownmenus with the names of the targets.

Declaration

```
public async void GenerateTargetItems()
```

GetCurrentlySelectedTarget(Int32)

Get the position data of currently selected target.

Declaration

```
public Vector3 GetCurrentlySelectedTarget(int selectedValue)
```

Parameters

Type	Name	Description
System.Int32	*selectedValue*	Value of dropdown to select a target.

Returns

Type	Description
Vector3	The Vector of the position data from the selectedValue provided.

makeallunselected()

If the user goes back to the menu all targets get unselected and their original color gets restored.

Declaration

```
public void makeallunselected()
```

removeTarget()

Remove a selected target from the environment.

Declaration

```
public void removeTarget()
```

restartpositions()

Deletes all changes made to the environment and loads the data from json again, in generel it reloads the last game settings.

Declaration

```
public void restartpositions()
```

SaveData()

Save all targets to the json to save the changes for the next users. Save the data to a string for uploading to the server.

Declaration

```
public void SaveData()
```

sendjson()

Save JSON to server and check if it worked.

Declaration

```
public IEnumerator sendjson()
```

Returns

Type	Description
System.Collections.IEnumerator	

SetSelectedChange2(Int32)

Notize when a target gets selected and set item like the value selected. Set the sliders to change the coordinates of the target selected.

Declaration

```
public void SetSelectedChange2(int selectedValue)
```

Parameters

Type	Name	Description
System.Int32	*selectedValue*	The value selected by the dropdown on third page.

SetSelectedChange3(Int32)

Notize when a target gets selected and set item like the value selected. Set the sliders to change the coordinates of the target selected.

Declaration

```
public void SetSelectedChange3(int selectedValue)
```

Parameters

Type	Name	Description
System.Int32	*selectedValue*	The value selected by the dropdown on third page.

SetSelectedTargetPositionWithDropdown(Int32)

Get the position of the target selected.

Declaration

```
public void SetSelectedTargetPositionWithDropdown(int selectedValue)
```

Parameters

Type	Name	Description
System.Int32	*selectedValue*	selectedValue by the dropdownmenu.

Start()

When the Game is started this method gets executed. It generates all targets, fills the dropdown menu with the targets and sets the paths were the json is stored.

Declaration

```
public void Start()
```

Update()

Method used to always check if the app is started. Checks if item is selected and always updates the position when slider gets moved. Changes color of the selected item from red to blue and backwards.

Declaration

```
public void Update()
```