Inloggen d.m.v. een QR-code

Pieter Barnhoorn

25-1-21

# Inleiding

HELLOBuurman wil dat het mogelijk is, voor een deelnemer van een bepaald traject, om in de app in te loggen d.m.v. een gegenereerde QR-code. De bedoeling is dat gebruikers een brief van de gemeente krijgen waar de QR-code in aanwezig is, en deze daarna scannen in de app, waardoor ze toegang krijgen tot de app en de locaties kunnen bekijken. Op dit moment wordt er ingelogd d.m.v. een code die gebruikers van de gemeente krijgen. Aan deze code zit een project vast, en door deze code wordt ook onderscheid gemaakt tussen gebruikers.

Alle benodigde bestanden zijn te vinden in de volgende Git repository: <https://github.com/PJJSB/UnityQRCodeReaderWriter>

# QR-code

Een QR-code (Quick Response Code) is een tweedimensionale streepjescode. Een QR-code wordt meestal gebruikt om data of links in op te slaan (Wikipedia). Het voordeel van een QR-code is dat gebruikers geen reeks aan karakters hoeft in te typen om in te kunnen loggen in de app. Dit kost tijd en voor mensen die moeite met zicht hebben, is dit al helemaal moeilijk. Het gebruik van een QR-code zorgt ervoor dat het inloggen een stuk sneller gaat en de gebruiker zelf een stuk minder hoeft te doen. Dit is dus ook een stuk gebruiksvriendelijker.

# Stappenplan inloggen met QR-code

1. De gebruiker opent de brief.
2. De gebruiker opent de app.
3. De gebruiker klikt op de ‘log in’ knop.
4. De camera opent in de app.
5. De gebruiker scant de QR-code uit de brief.
6. De app decodeert de QR-code.
7. De app leest de toegestuurde informatie uit.
8. De app laadt het juiste project in en slaat het meegegeven user-ID op, zodat deze gebruikt kan worden bij het plaatsen van commentaar.
9. De gebruiker komt bij het eerste shot van het project terecht.

# Risico’s

Er zijn natuurlijk risico’s verbonden aan het gebruik van een QR-code als inloggegeven, dit zijn:

* **Onderscheppen van de brief**

De brief die de gemeente stuurt naar de bewoners, kan onderschept worden door iemand die niet mee mag/ kan doen met het project. Dit heeft als gevolg dat er stemmen en opmerkingen kunnen komen van mensen die helemaal niet in de buurt wonen en dus geen zeggen zouden mogen hebben over wat er in de wijk gaat gebeuren.

* **Het veranderen van de QR-code**

Als de opgestuurde brief onderschept zou worden, is er een kans dat de QR-code vervalst zou kunnen worden en de gegevens van de code veranderd. Deze zou kunnen leiden naar bijvoorbeeld een geïnfecteerde website

Deze risico’s zijn ook aanwezig als er geen QR-code gebruikt wordt, dus zijn deze niet QR-code specifiek.

# QR-code scannen in Unity3D

Het scannen van een QR-code in Unity3D is volgens de blogpost van A. M. Nenu uit 2016 erg makkelijk. Hij maakt gebruik van een library genaamd ”Zxing”. Dit is een open-source, multi-format image processing library (Zxing). Deze library bevat code om verschillende vormen van barcodes te encoden en te decoden (dus ook QR-codes). Het maken van een QR-code is met deze library ook erg simpel, dus mocht de app ooit een QR-code aan moeten kunnen maken, is dit mogelijk d.m.v. deze library.

# Testproject opzetten

In het bijgaande project genaamd ‘QRCodeReader’ is een klein voorbeeld gemaakt waarin een app de camera opstart, en vervolgens de QR-code van het scherm leest. Dit project zit samen met de package in dezelfde folder als dit document.

Voordat je kan beginnen moet je de juiste DLL aan het project toevoegen, zodat je de functionaliteit van deze package kan gebruiken. Dit doe je door in Visual Studio in de menubalk ‘Project’ aan te klikken, en daarna op ‘Add Reference’ te klikken. Daarna zoek je naar de bijgeleverde DLL genaamd ‘zxing.unity.dll’ en voeg je deze toe. Na deze stappen liep ik tegen het probleem aan dat Unity een reference naar de assembly mist. Dit heb ik opgelost door de correcte files te downloaden van de laatste release versie en deze in Unity3D toe te voegen in een mapje, hier een assembly definition naast te zetten, en een assembly te maken naast het script. In de assembly naast het script heb ik daarna een referentie naar de assembly van de package gezet, en daarna werkte het.

Daarna heb ik geprobeerd te builden, maar kreeg ik de error *“UnityException: Build path contains project built with "Create Visual Studio Solution" option, which is incompatible with current build settings. Consider building your project into an empty directory.”.* Een lege map aanmaken en daarin builden heeft het probleem opgelost, en er kan nu een build gemaakt worden.

Na het builden heb ik ervoor gezorgd dat het lezen van een barcode om de seconde geroepen wordt, i.p.v. ieder frame, omdat dit inhakt op de performance. Daarna werkt het scannen zelf en kan het script een QR-code decoden.

Het enige probleem waar ik nu tegen aan loop is dat het scherm 90 graden met de klok mee gedraaid moet worden. Dit heb ik opgelost door “*GUIUtility.RotateAroundPivot(90f, new Vector2(Screen.width/2, Screen.height/2));”* aan de functie toe te voegen die de camTexture op het scherm tekent.

# QR-code genereren

De ZXing library kan niet alleen QR-codes uitlezen, maar ook genereren. In het testproject heb ik een nieuw script aangemaakt waarin een aantal QR-codes gegenereerd worden om zo te kunnen testen of dit gebruikt kan worden in de app. Dit kan heel makkelijk gedaan worden door een stukje tekst aan de ‘writer’ mee te geven en de Color32[] op te vangen in een texture. Deze texture kun je daarna omzetten naar jpg of png formaat, en deze naar een bestand schrijven.

# Bronnen

Wikipedia, 11-12-2020, “QR-code”, geraadpleegd van: <https://en.wikipedia.org/wiki/QR_code>

A. M. Nenu, 27-10-2016, “Reading and Generating QR codes with C# in Unity 3D — the easy way”, geraadpleegd van: <https://adriannenu.medium.com/reading-and-generating-qr-codes-with-c-in-unity-3d-the-easy-way-a25e1d85ba51>

Zxing, 30-9-2020, “ZXing ("Zebra Crossing") barcode scanning library for Java, Android”, geraadpleegd van: <https://github.com/zxing/zxing>