**AR escape room**



Authors:

    Mikk Õunmaa, s2909650

    Wytze Breuke, s2991918

# System requirements

## System requirements for running the app:

* Android phone with at least a version  8.1
* ARCore supported device,  <https://developers.google.com/ar/discover/supported-devices>
* App needs to have access to the camera.
* Tested with Samsung 9, and Redmi Note 7.

## System requirements for building the app:

* Unity version 2019.4.14f1
* JDK version >= 1.8
* Gradle version >= 6.7 and Unity configured to use it
* Unity has Android development tools installed
* In Unity package manager, AR Foundation, ARCore XR Plugin, ARKit Face Tracking and ARKit XR Plugin installed with version 4.0.8. TextMesh Pro also needed.

# Authorship of assets

**Following sections contains spoilers**

## Created by us:

* 3D models:
  + Gold
  + Added “Caesar” name to the Caesar Bust
  + End portal with the room
  + Cookie and the text inside the cookie
  + 3d Texts on Monarchs
  + Numbers
* Pictures:
  + Pictures of playing cards
  + Pictures of countries (taken from the bosatlas)
  + The actual maze generated online (http://www.mazegenerator.net/)  and merged with the poker chip
  + Text on boarding passes
* Code base:
  + All script files are written by us besides TextAlert
  + Shaders under Custom section are copies from the premade shaders with little additions made by us to make them work with the portal
* Sound:
  + Morse code generated online
* Texts:
  + Introduction
  + Texts about monarchs are taken from Wikipedia, with little modifications according to our needs.
  + Monarch room introduction
  + Hints
* Video
  + Video playing on face

## Created by others:

* 3D models:
  + Plane
  + Caesar bust
* Sound effects:
  + Plane
  + Police
  + Cookie
  + Unlocking sound
* Pictures
  + Phone
  + Clock
  + Chicken picture
  + Lock
  + Background of Maze
  + Base of boarding passes
  + Pictures of monarchs
* Textures
  + Home screen background
  + Button texture

# Source code

All the used scripts are written by us besides TextAlert. TextAlert is used to show the dialog message to the player when the time runs out.

# Overview of the system:

It is an AR Escape room. It uses image recognition to show 3D objects on the images. To have full experience of the game you also need some physical equipment like locks, deck of cards and world atlas.

Pictures and their function:

Cards and countries only work with a specific deck of cards and atlas.

* Change camera between front-facing and world facing.
* Front-facing camera shows video on face

All images are tracked with the world facing camera.

* Clock – Shows timer on screen
* Phone – Plays morse code
* Chicken – Plays chicken sound
* Cookie – Shows a cookie, when the cookie is visible, click on it (“to eat” or “open it”) and it will show a quote. When clicked the first time, it plays a sound effect. The quote is shown now every time the cookie picture is shown to the camera.
* Lock – Shows an overlay with buttons, when right directions are entered, plays unlock sound effect.
* Maze – Shows a riddle text, after right directions have been entered to the lock
* Elizabeth II – Shows “Name”
* Henry VII – Shows “Key”
* James VI – Shows “Is”
* William I – Shows “Cipher”
* Louis XIV – Shows Caesar’s Bust
* Jack of Spades – Shows a number
* Queen of Hearts – Shows a number
* King of Diamonds – Shows a number
* Countries in the following order:
  + Netherlands – Shows a plane and plays plane sound effect
  + Japan – Shows a plane and if plane sound effect not playing, plays it.
  + South Africa – Shows a plane and if plane sound effect not playing, plays it.
  + Morocco – Shows a portal with gold and time taken to complete, and if plane sound effect not playing, plays it. It should be possible to enter the portal, but it disappears sometimes.

# Instructions:

Install the .apk file on the Android phone, which supports AR Core. Give camera permission to the app. Start playing, the instructions on how to complete the game are in Walkthrough.pdf.  You can find the .apk file and the printable puzzle parts in the git repository

# How to compile the source code

### 1. Download the source code

First download the source code from git:

<https://github.com/Timpsik/augmentedReality>

### 2. Setup Unity

Install Unity Hub from: <https://unity3d.com/get-unity/download>

Go to <https://unity3d.com/get-unity/download/archive>

Select tab Unity 2019.x:

Then install Unity version 2019.4.14 using the Unity Hub button.

You might have to disable AntiVirus (in my case, AVG did not allow it), otherwise, it might block Unity from installing correctly Android NDK and Android SDK.

Check that the Android Build Support is also ticked when starting to install the version.

Startup Unity and open the folder augmentedReality from the git download.

Next, we need to configure Unity so that it can build AR Foundation apps.

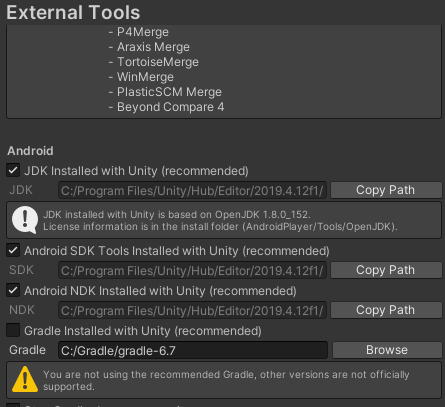
During the installation Unity should have installed these software packages:

* JDK
* Android SDK tools
* Android NDK
* Gradle

Unfortunately, the inbuilt Gradle version will not work, so you need to download your own Gradle version.

<https://gradle.org/next-steps/?version=6.7.1&format=bin>

The next step is to point Unity to the new Gradle version in Edit -> Preferences -> External tools. First uncheck the box Gradle installed with Unity, then use browse to select your installed Gradle version. If everything is installed correctly the External tools window will be looking like this:



The next step is to add the required plugins to Unity, this can be done in Unity by selecting Window-> Package Manager. Then install the following packages:

* AR Foundation,
* ARCore XR Plugin,
* ARKit Face Tracking and
* ARKit XR Plugin installed with version 4.0.8.
* TextMesh Pro

### 3. Setup of Android phone

This concludes the configuration of Unity, next we need to configure the phone upon which the app will be installed. First check if the phone has support of AR core:

<https://developers.google.com/ar/discover/supported-devices>

Next step is to allow USB debugging on your phone this differs for some phone models so please follow the instructions written here:

<https://developer.android.com/studio/debug/dev-options>

Connect the phone to the Computer using USB.

### 4. Compiling the source code and running it on the phone

If everything was set up correctly it should now be possible to compile the source code. This can be done by using File -> Build and Run in Unity.

If there are any problems, you can get in touch by sending an email to:

[w.p.breukel@umail.leidenuniv.nl](mailto:w.p.breukel@umail.leidenuniv.nl)

[m.ounmaa@umail.leidenuniv.nl](mailto:m.ounmaa@umail.leidenuniv.nl)