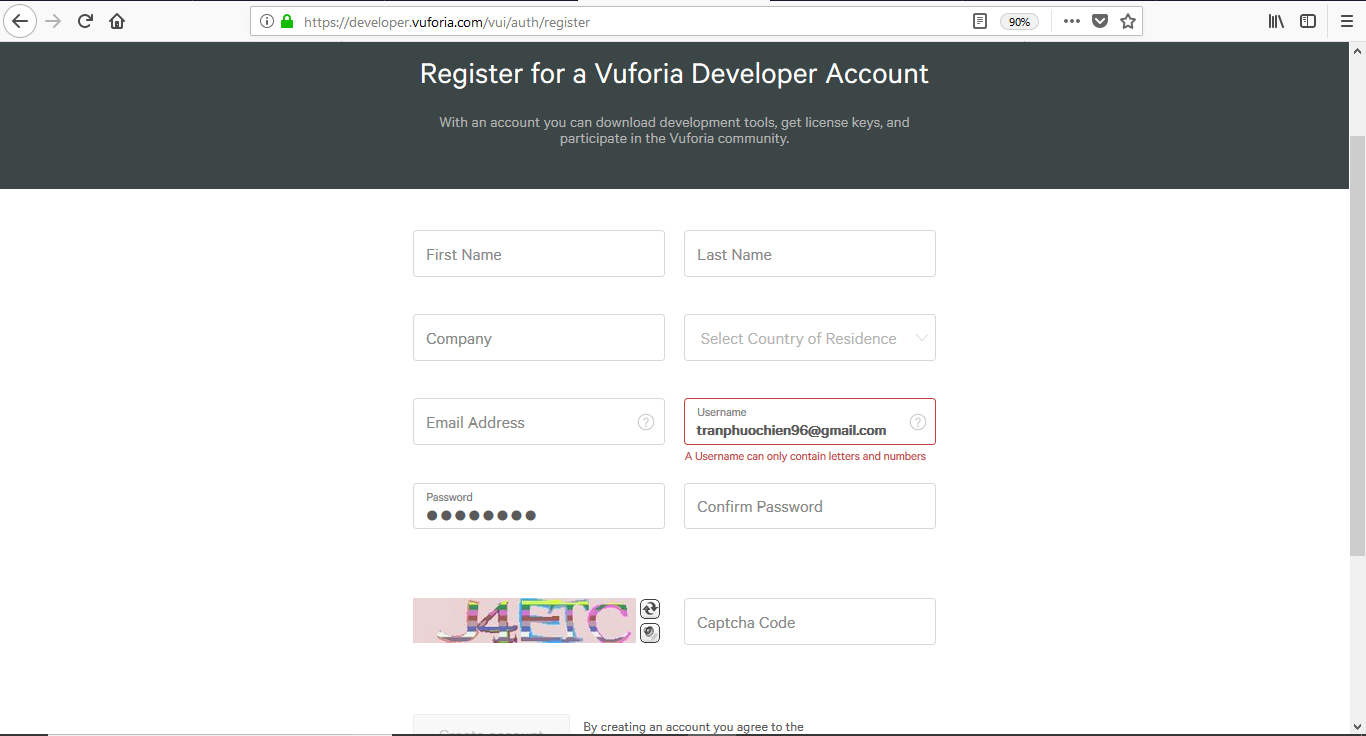
1. Sử dụng Vuforia trong Unity

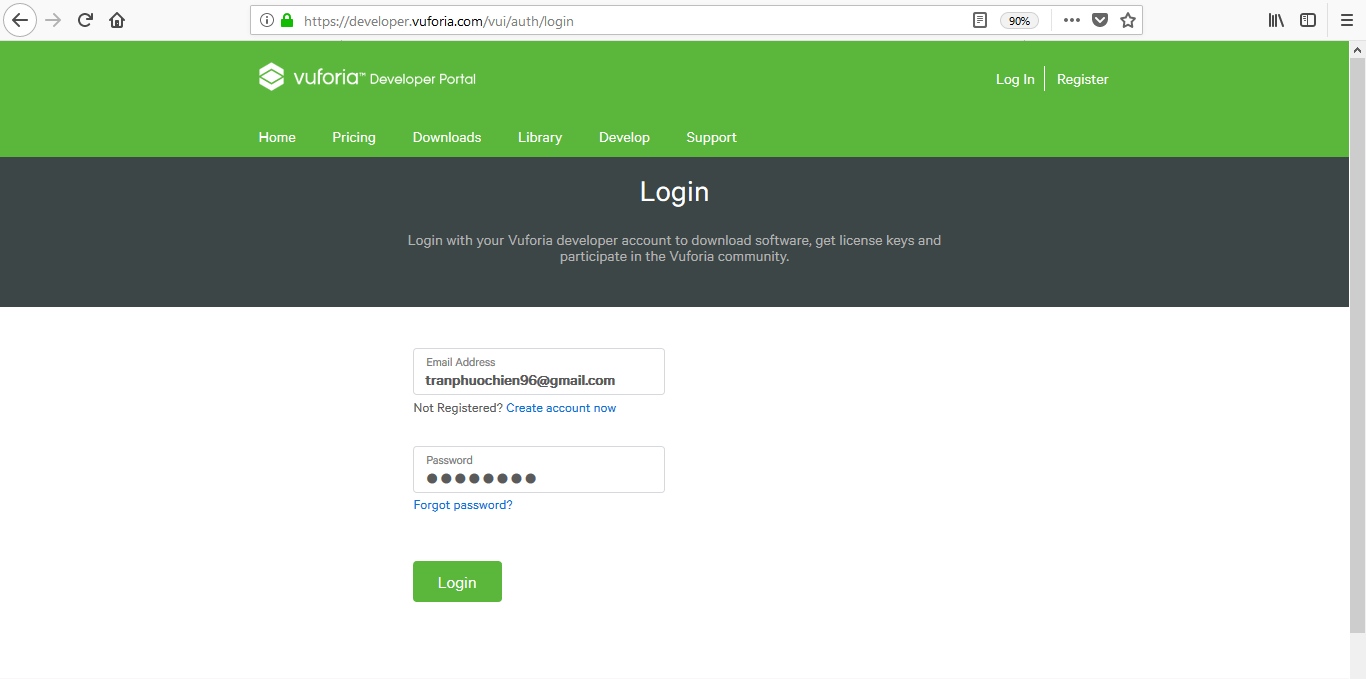
Với mục đích sử dụng Vuforia để nhận diện Object trong trường hợp vật cần nhận diện có nhiều họa tiết hoa văn chứ không phải là một mặt phẳng đơn sắc. Chúng em tiến hành sử dụng Vuforia để kiểm tra mức độ hiệu quả của nó so với nhận diện bằng Blob.

1. Đăng kí tài khoản sử dụng Vuforia API

Đầu tiên, truy cập vào <https://developer.vuforia.com/vui/auth/register> để đăng kí tài khoản developer. Sau khi điền đủ thông tin vào mẫu đăng kí và xác nhận tạo tài khoản, ta xác thực việc đăng kí với email được gửi đến.

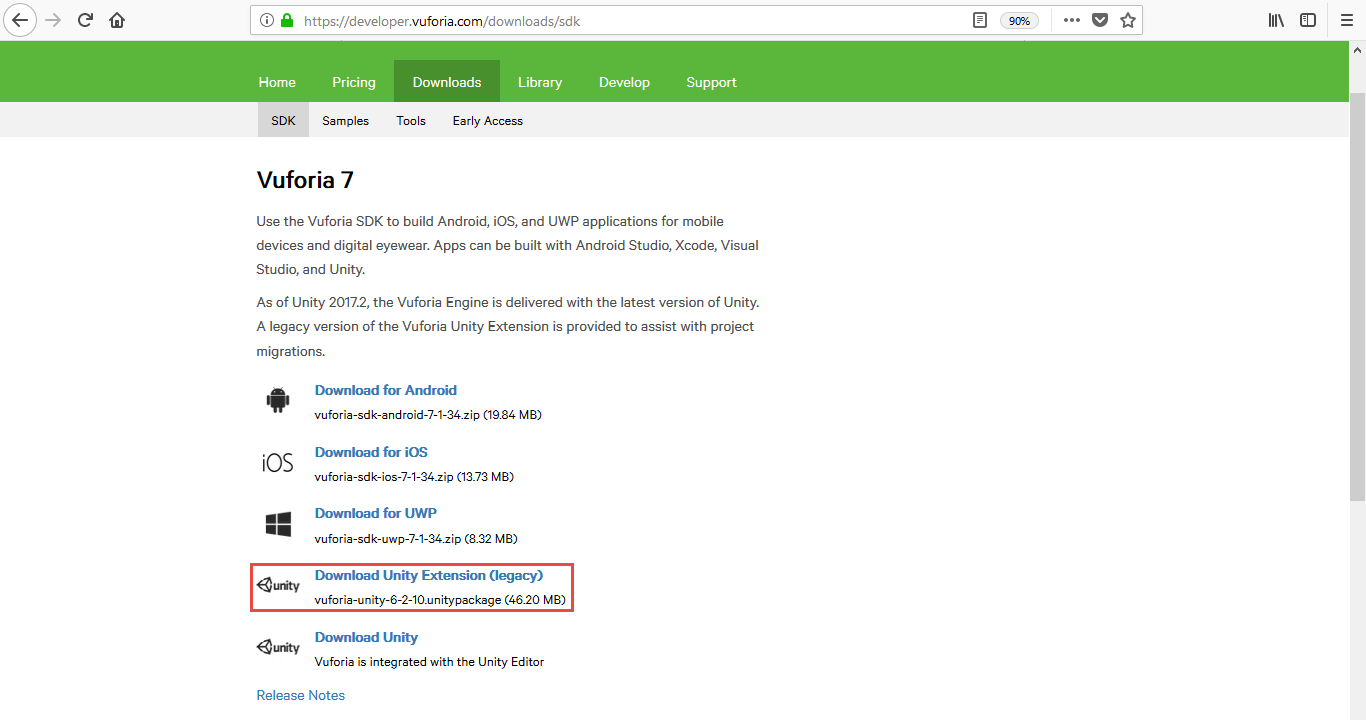


Sau khi xác nhận tài khoản, ta đăng nhập vào hệ thống với đường dẫn: <https://developer.vuforia.com/vui/auth/login>

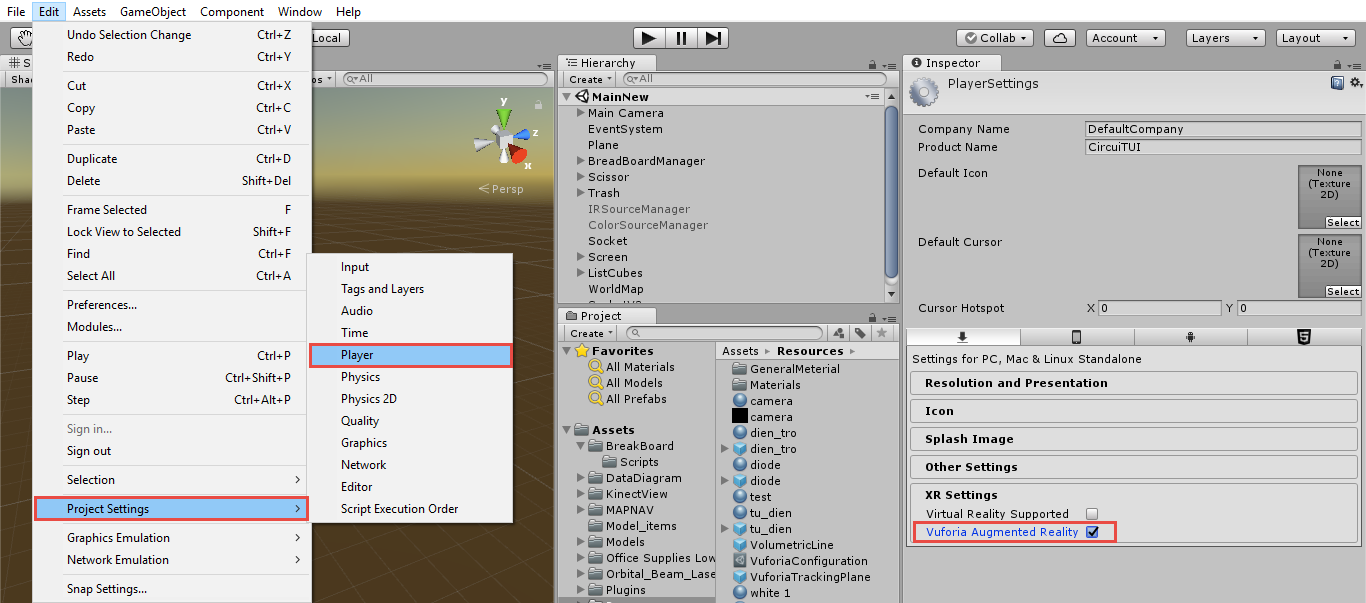


1. Sử dụng Vuforia API với Unity

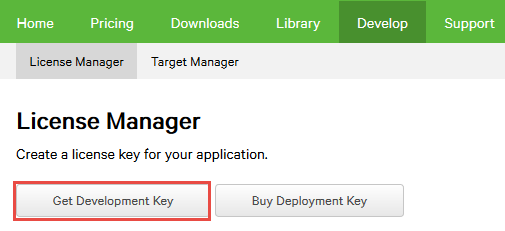
Trước hết, ta cần tải bộ cài từ trang Download của Vuforia với đường dẫn sau: <https://developer.vuforia.com/downloads/sdk>. Cài đặt được tải ở dạng là một package của Unity, khi tải xong sẽ được cài trực tiếp vào Unity. Tuy nhiên đây là cách cài đặt dành cho Unity có phiên bản trước 2017.2, từ phiên bản 2017.2 trở về sau, Vuforia có thể được cài đặt với các package khác khi cài đặt Unity.

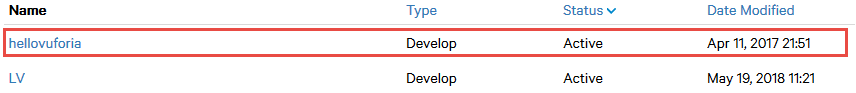


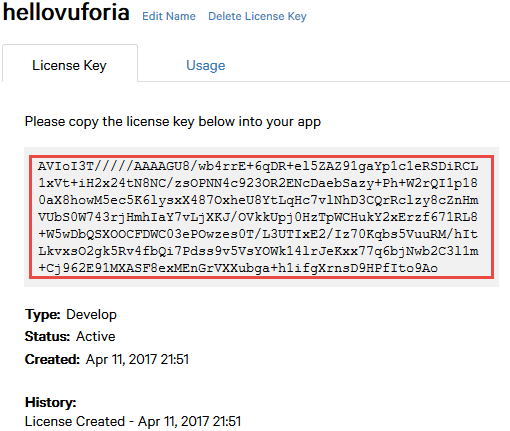
Tiếp theo, ta khởi động Unity, Chọn Edit => Project Settings => Player. Ở cửa sổ Inspector, ta đánh dấu vào “Vuforia Augmented Reality”.



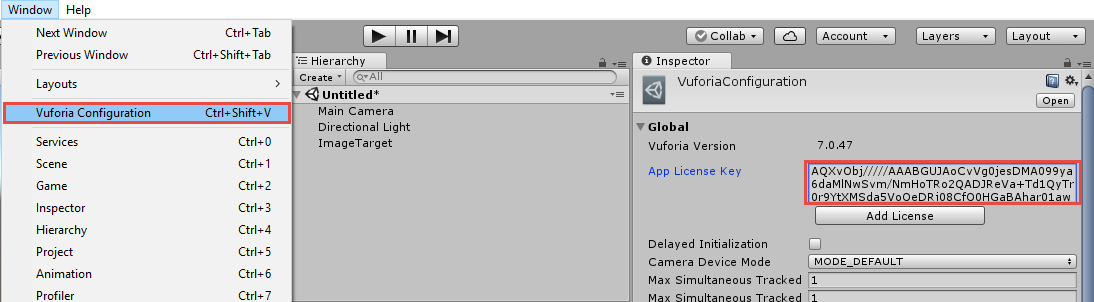
Để sử dụng các chức năng của Vuforia thì ta chọn Game Object => Vuforia rồi chọn chức năng tương ứng, ở đây chúng em lấy chức năng cơ bản của Vuforia là Image Target. Để sử dụng Image Target thì ở trang developer của Vuforia ta cần tạo một App Key để sử dụng. Khi điền đủ thông tin và xác nhận tạo key, ta có một App Key được tạo như hình dưới, ta nhấn vào App Key Name để lấy Key.



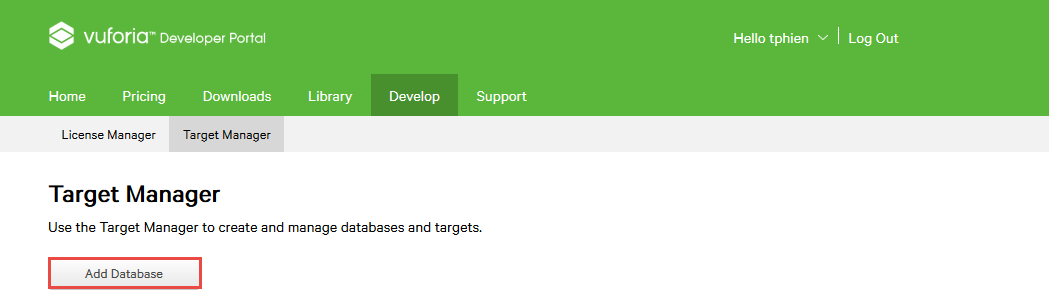


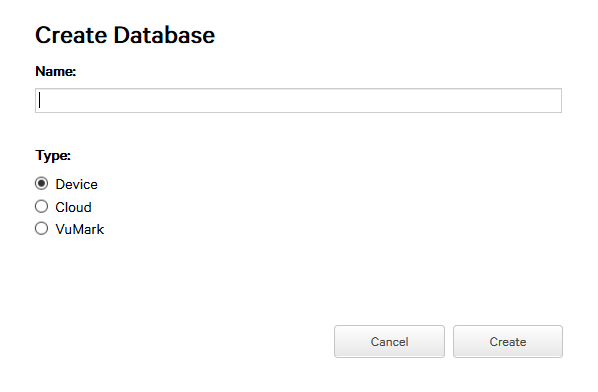


Ta copy phần App Key, ở màn hình Unity, ta chọn Window => Vuforia Configuration, sau đó Key vào ô App License Key.

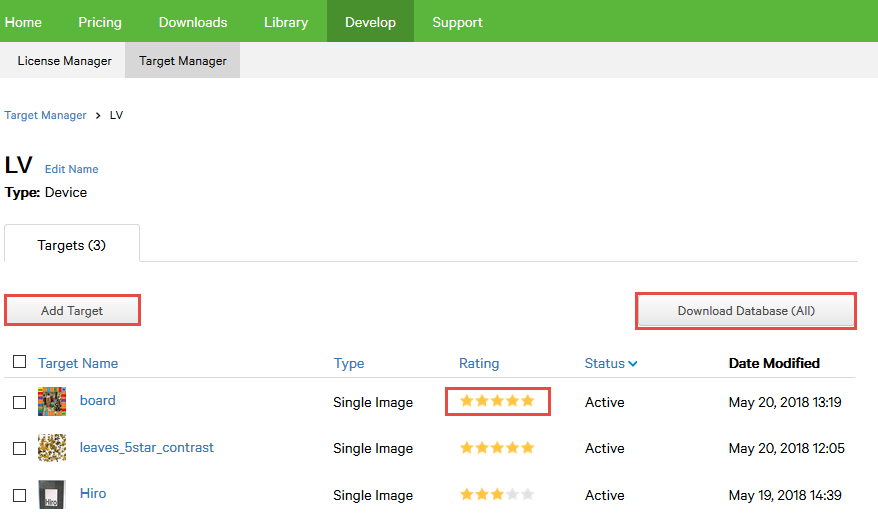


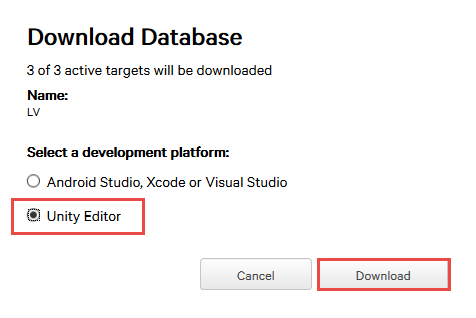
Bây giờ, ta đã có thể sử dụng Image Target, chức năng cơ bản của Image Target là nhận diện một hình ảnh 2D đã được thêm vào database của Vuforia. Để thêm một hình ảnh vào data base ta đăng nhập vào trang chủ Vuforia, ở Tab Develop chọn Target Manager và nhấn Add Database. Đặt tên Database và tạo với type là Device.





Ta nhấn vào tên Database và nhấn Add Target để thêm dữ liệu vào Database. Có thể thêm hình ảnh 2D hoặc các vật thể 3D. Sau khi upload dữ liệu, điền kích thước và đặt tên thì ta nhấn Add để xác nhận. Khi dữ liệu được đưa vào Database thì mức độ đánh giá khả năng nhận diện được thể hiện qua phần Rating. Cuối cùng nhấn vào Download Database (All) để tải về Unity. Khi tải về ta chọn Development platform là Unity Editor rồi nhấn Download. File tải về dưới dạng là package của Unity sẽ được cài trực tiếp.



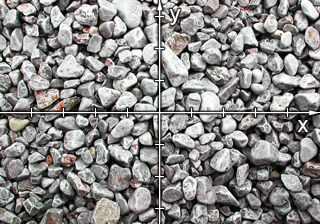
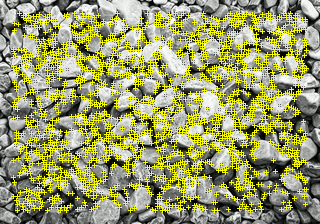


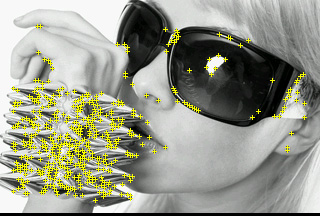
Ở giao diện Unity, ta chọn Database tương ứng vừa tải về và chọn Image Target là ảnh được đưa vào Database. Sau đó là có thể Start Game Mode của Unity và nhận diện.

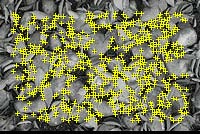


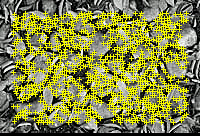
1. Tiêu chí Rating ảnh của Vuforia và các yếu tố ảnh hưởng đến việc nhận diện.

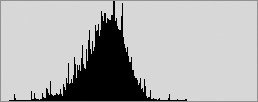
Vuforia có thể nhận diện và theo dấu Targets bằng việc phân tích sự tương phản dựa trên những đặc tính được rút trích từ Target đó thể hiện cho camera như họa tiết của ảnh, tỉ lệ ảnh, chất lượng ảnh, độ cân bằng trong phân bố các chi tiết ảnh… Có thể cải thiện hiệu quả nhận diện bằng cách cải thiện những đặc tính đó (tăng độ chi tiết, độ tương phản màu sắc tốt, không bị lặp nội dung, chi tiết có nhiều góc cạnh). Format của ảnh phải là ảnh PNG hoặc JPG 8 hoặc 24 bit, nhỏ hơn 2MB, với ảnh JPG phải dùng hệ màu RGB hoặc greyscale (không sử dụng hệ màu CMYK). Để đánh giá tiêu chí Rating với ảnh Greyscale thì ta dựa vào Greyscale Histogram, ảnh có độ tương phản cao, có Histogram rộng và phẳng thường là ảnh có Rating cao.

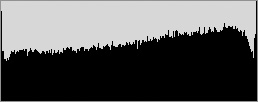
 

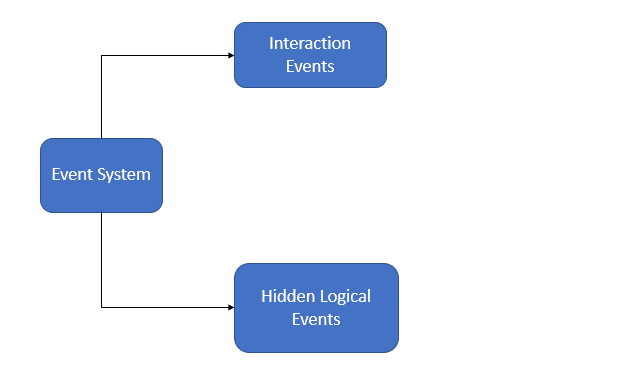




Về camera thì nếu có chế độ focus sẽ tốt hơn cho việc nhận diện. Bên cạnh đó, điều kiện ánh sáng môi trường cũng có ảnh hưởng lớn đến việc nhận diện và theo dấu. Vuforia hoạt động tốt nhất với ánh sáng trong phòng vì nó ổn định và dễ dàng điều chỉnh cường độ sáng. Kích thước Target tốt nhất từ 5 inches hoặc 12cm trở lên về chiều rộng cùng với kích thước hợp lí theo tỉ lệ ảnh về chiều cao. Khoảng cách từ ảnh đến camera trong điều kiện hoàn hảo khoảng bằng 10 lần chiều rộng của ảnh. Ảnh được in ra cần đảm bảo độ phẳng, không bị bẻ cong hay nhăn, nên sử dụng giấy bìa cứng để in ảnh nhận diện. Bề mặt ảnh với camera nên là bề mặt nhám, vì bề mặt bóng dưới nguồn sáng và góc chiếu sáng ngẫu nhiên có thể dẫn đến hiện tượng phản xạ, khúc xạ làm mất đi chi tiết của ảnh. Góc nhìn của camera đến ảnh nên trực diện và tránh góc nhìn quá hẹp so với bề mặt ảnh nhận diện.

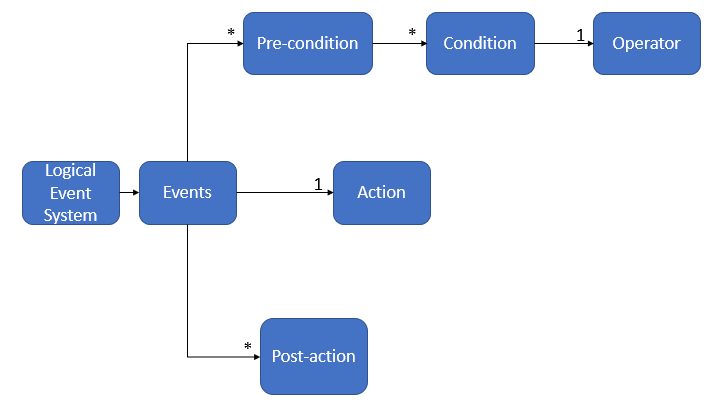
Sau quá trình thử nghiệm với Vuforia, chúng em thấy được lợi điểm của nó cho phép nhận diện và theo dấu được vật có bề mặt phức tạp. Tuy nhiên, nó cũng bị phụ thuộc vào điều kiện môi trường như Blob và khả năng mapping tọa độ thu được của vật trong thế giới ảo với vật trong thế giới thật.

1. Event System

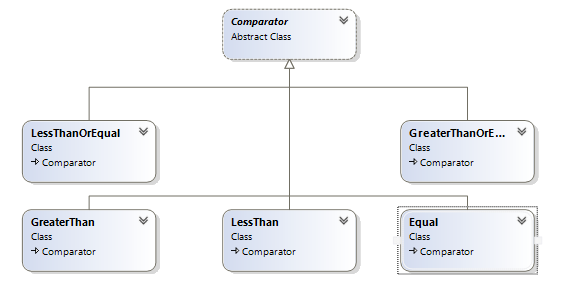
Hệ thống sự kiện (Event System) được thiết kế với mục đích quản lí kịch bản hoạt động của hệ thống một cách hiệu quả và tách biệt các thành phần của hệ thống với nhau để chúng hoạt động độc lập và dễ dàng thay đổi, mở rộng. Các thành phần ở đây là việc xử lí một sự kiện xảy ra, và việc kiểm tra điều kiện để kích hoạt sự kiện đó. Nó cho phép việc load nhanh một kịch bản và có thể thay đổi kịch bản đó một cách linh hoạt mà không phụ thuộc vào kiến trúc xử lí sự kiện bên trong. Bên cạnh đó hệ thống sự kiện cho phép việc kích hoạt liên hoàn sự kiện một cách dễ dàng. Hệ thống sự kiện không chỉ có kịch bản sự kiện ẩn liên hoàn mà còn bao gồm cả các sự kiện và việc xử lí sự kiện do người dùng tạo ra (như cách xử lí sự kiện thông thường). Ở đây chúng em tập trung vào việc xử lí liên hoàn sự kiện theo kịch bản. Minh họa hệ thống sự kiện như sau:

1. Các thành phần của hệ thống sự kiện liên hoàn gồm:

* Một mảng các đối tượng Event, mỗi Event tương ứng với một bộ các hành động tác động lên các Object tùy thuộc vào kịch bản.
* Một Dictionary chứa các cờ hiệu (Flags), trong quá trình hoạt động, chương trình sẽ duyệt Dictionary này một cách xuyên suốt. Nếu có một tổ hợp cờ nào phù hợp với một Event tương ứng thì Event đó sẽ được kích hoạt.
* Một Dictionary các Object sẽ bị tác động trong kịch bản, ở đầu chương trình sẽ tiến hành đăng kí các Object này vào Dictionary.
* Một kịch bản đầu vào ở dạng JSON File, mô tả các Event. Cấu trúc gồm tên của Event, Action, Pre-conditions, Post-conditions.s
* Minh họa:



1. Cấu trúc của một Event:
   * Pre-conditions: Tập hợp các điều kiện đầu vào, một Pre-condition gồm các Conditions. Sự kiện được kích hoạt khi một Pre-condition thỏa.
   * Conditions: Là một chuỗi các điều kiện được nối với nhau bằng toán tử &&, một Pre-condition được kích hoạt khi chuỗi các điều kiện đó đều thỏa.
   * Comparators: Là các toán tử so sánh dùng để so sánh các cờ hiệu với một giá trị nào đó, nhằm kích hoạt Conditions từ đó kích hoạt Pre-condition và cuối cùng là kích hoạt sự kiện. Các toán tử này bao gồm: Equal, GreaterThan, LessThan, LessThanOrEqual, GreaterThanOrEqual. Các toán tử này kế thừa Abstract Class Comparator và được tạo ra một lần duy nhất ở đầu chương trình để sử dụng. Nói cách khác đây là một Global Helper Class.
     + Minh họa:



* + - Code mình họa một Comparator:

public class GreaterThanOrEqual : Comparator

{

public override bool compare(string FlagId, int value)

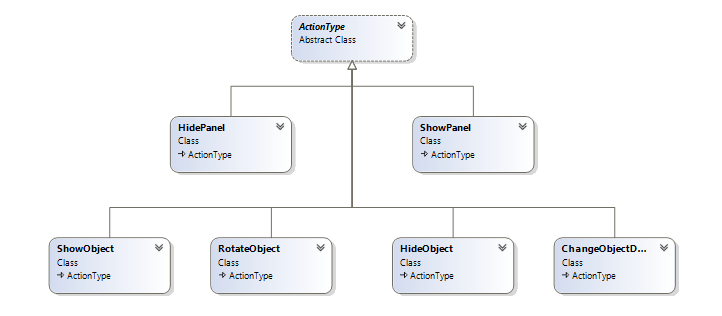
{

return Global.getFlag(FlagId) >= value;

}

}

* Action: Là một hành động sẽ tác động lên một GameObject trong hệ thống khi sự kiện được kích hoạt (ví dụ như ẩn, hiện, di chuyển vị trí, thu nhỏ, phóng to,…). Các class này kế thừa class Action Type là Base Abstract Class cho các Action này
  + Minh họa:



* + Code mình họa một Action:

public class RotateObject : ActionType

{

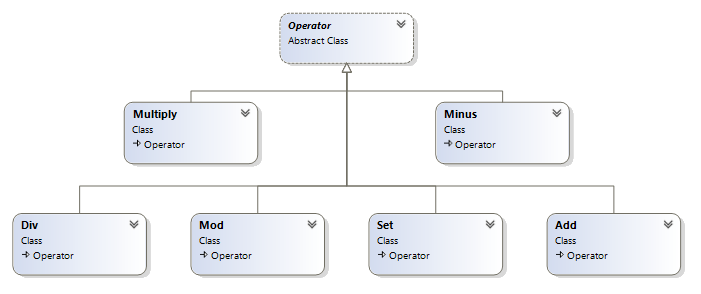
public override void DoAction(GameObject gameObject, ParamsRaw paramsRaw)

{

gameObject.transform.Rotate(0, 0, 30 \* Time.deltaTime, Space.World);

}

}

* Post-actions: Là chuỗi các hành động dùng để cập nhật các phần tử của Dictionary Flags sau khi Action của Event được thực hiện. Đây là cơ chể để thực hiện việc kích hoạt liên hoàn sự kiện trong hệ thống. Vì Dictionary Flags sẽ luôn được thăm qua mỗi lần lặp Update, dựa vào trạng thái cờ mà sẽ kích hoạt sự kiện, nếu một sự kiện được kích hoạt, sau đó Post-actions cập nhật lại Dictionary Flags tạo ra một tổ hợp trạng thái mới kích hoạt một sự kiện khác, từ đós tạo ra hiệu ứng liên hoàn.
* Operator: Mỗi hành động trong Post-actions là một Operator (cộng, trừ, nhân, chia, gán,…) tác động đến giá trị của Dictionary Flags.ss
  + Minh họa:
  + Code mình họa một Operator:

public class Mod : Operator

{

public override int Operate(string FlagId, int value)

{

return Global.setFlag(FlagId, Global.getFlag(FlagId) % value);

}

}

1. Cấu hình hệ thống
   * + 1. Thiết kế:

Hệ thống của chúng em bao gồm một khung bàn nhôm có chiều rộng 45cm và chiều dài là 73cm, bàn cao 103cm. Mặt bàn thì chỉ có viền cạnh và rỗng bên trong. Đặt phía trên khung là một tấm kính trong suốt và phía trên tấm kính là giấy trắng để hứng ảnh. Chúng em sử dụng máy chiếu Sanyo PLC-XW60 với cường độ sáng 2000 Ansi Lumens, độ phân giải 1024x768, kích thước hình ảnh từ 80 đến 300 inches. Máy chiếu theo góc 90 độ hướng lên từ dưới chân bàn lên mặt kính trong, hình ảnh sẽ được giữ lại ở mặt giấy trắng, để hình ảnh thuận chiều với mắt người nhìn từ phía trên thì chúng em tiến hành lật màn hình phát ra từ máy chiếu theo chiều dọc. Với máy chiếu trên có cảm biến về góc chiếu nên chúng em đã điều chỉnh keystone để hình ảnh phát ra là hình chữ nhật với góc người nhìn, cuối cùng là điều chỉnh độ thu phóng và focus của thấu kính. Chúng em dùng camera Logitech C170 có khả năng quay video 1024x768, ảnh lên tới 5 megapixel, USB 2.0 để thu hình từ phía trên. Camera được lắp cao cách 75cm so với mặt bàn, có góc nhìn sao cho lấy đủ mặt bàn và hình ảnh hứng được. Camera được dùng để thu ảnh và nhận diện vật thể trong quá trình sử dụng. Để hình ảnh phát ra hứng được là rõ nhất thì ngoài việc chỉnh độ focus của thấu kính chúng em có 2 hướng giải quyết: Một là giảm ánh sáng môi trường bên ngoài tác động vào mặt bàn – đây là cách giải quyết chúng em đang thực hiện – với mức sáng hiện tại chúng em đang sử dụng là 4 lux trở xuống (ánh sáng phòng thí nghiệm ban ngày không mở đèn phòng và đóng rèm là 8 lux). Hai là thay đổi chất liệu giấy hứng ảnh để thu được hình ảnh tốt hơn, nhưng hiện tại chúng em chưa tìm ra được chất liệu giấy phù hợp. Cuối cùng là một thiết bị di động để hỗ trợ tương tác trong quá trình hoạt động của hệ thống, ở đây chúng em sử dụng điện thoại Xiaomi có màn hình 5.5 inches Full HD, hệ điều hành Android 7.0, CPU Snapdragon 625 8 nhân 2.0 Hz, RAM 3 GB, chip đồ họa Adreno 506 và máy tính bản Huawei Mediapad T3 màn hình 8 inches, độ phân giải 1280x800 pixels, hệ điều hành Android 7.0, CPU Qualcomm MSM8917 1.4 Hz, RAM 2 GB, chịp đồ họa Adreno 308.





