GESTURE RECOGNIZER DOCUMENTATION

V3.0

LAST UPDATED: 19.06.2016

Table of Contents

WHAT IS GESTURE RECOGNIZER?

Gesture Recognizer is a very simple tool for handling and recognizing user input. Users can draw on the screen and Gesture Recognizer can recognize the sketch.

HOW DO I USE GESTURE RECOGNIZER?

This version of the Gesture Recognizer has been developed from ground-up. The functionality is still the same, but usage is easier than ever.

Gesture Recognizer relies on events when dealing with a captured gesture. All you need to do is to drag & drop the **GestureRecognizer** prefab onto the scene and subscribe to its recognition events:

1- Drag & drop the **GestureRecognizer** prefab onto the scene from the **Prefabs** folder. Modify it as you like (color, width etc.). Don't forget to fill **Library To Load** attribute on this prefab. This is the name of the gesture library (see: How Do I Add My Own Gestures?) that includes your gestures.

2- Create a C# script and include the `GestureRecognizer` namespace at the beginning of your script:

```
using GestureRecognizer;
```

3- Add your own method and name it as you desire. This is the method to do whatever you want to do (shoot fireballs, cast magic spells etc.). For the sake of clarity, I named it `OnGestureRecognition` here:

```
void OnGestureRecognition(Result r) {
    Debug.Log("Gesture is " + r.Name + " and scored: " + r.Score);
}
```

4- Subscribe this method to `OnRecognition` method of `GestureBehaviour` in `OnEnable` and unsubscribe it in `OnDisable` and `OnDestroy`:

```
void OnEnable() {
    GestureBehaviour.OnRecognition += OnGestureRecognition;
}

void OnDisable() {
    GestureBehaviour.OnRecognition -= OnGestureRecognition;
}

void OnDestroy() {
    GestureBehaviour.OnRecognition -= OnGestureRecognition;
}
```

HOW DO I ADD MY OWN GESTURES?

The collection that contains gestures is called a "gesture library" and Gesture Recognizer includes 3 libraries by default: **shapes**, **alphabet** and **numbers**. You can edit these libraries or create your own. Creating your own gesture library is very easy:

1- Right click somewhere empty in **Project** window and choose **Create** -> **Gesture Library**. Name it as you like and press enter. This will create an empty gesture library.

2- Click on the library you just created to select it and click **Open in Gesture Editor** in **Inspector** window. This will open the **Gesture Editor** window which lets you add/remove or rename gestures.

3- Right click somewhere empty in **Gesture Editor** window and click **Add gesture**. Draw your gesture in the highlighted area, name it in the field below and click **Add**.

4- Click on the name of the gesture you just created and right click to rename or remove it.

5- Repeat step 3 to add more gestures.

In order to recognize a gesture, Gesture Recognizer compares it to all of the gestures in the library, scores them based on their similarity and returns the highest scoring gesture as the result. In order to increase the recognition accuracy, it is recommended to add at least 3 samples of the same gesture (because no one can draw a shape exactly the same twice, and sometimes it is possible to draw, let's say, a rectangle which looks more like a circle than an actual rectangle). You can add even more, but as the library grows, recognition performance would decrease. You need to find a balance between performance and recognition accuracy.

HOW DO I LIMIT DRAWING TO A SPECIFIC AREA?

You can limit drawing a gesture to a specific area by changing **Gesture limit type** from **Unlimited** to **Start In Area**, **Clamp To Area** or **Ignore Outside**. After you select one of these options, find the child object with the name **Limited Draw Area**, and modify it as you like. Your gestures will be limited to this UI object according to chosen limitation option:

- **Start In Area:**

  Users will be able to start drawing only in this **Rect** area and continue to draw as they like.

- **Clamp To Area:**

  Users will be able to draw only in this **Rect** area and the points outside of it will be clamped to this area.

- **Ignore Outside:**

  Users will be able to draw only in this **Rect** area and the points outside of it will be ignored.

TROUBLESHOOTING

Problem: It does not recognize accurately! I drew a triangle but it says "it is recognized as rectangle". What gives?
Solution: The more you teach, the better GestureRecognizer recognizes. I suggest **adding at least three samples** for a single gesture. But beware; GestureRecognizer uses strings as keys for gestures. That means you need to write the correct name while adding a new gesture. If you draw a rectangle and name it as a triangle you might get confusing results.

Problem: Recognition accuracy did not get better even I have added more gestures. Any other suggestions?
Solution: Gesture Recognizer samples your gestures so that they have a certain amount of points, which is by default 64 (you can find this variable in **`Gesture.cs`** with the name `numberOfPoints`). As you draw more complex gestures, you will need a higher number of points, which would increase recognition accuracy. However, there are two downsides changing the number of points: a- performance of the recognition will be affected, b- you will have to resample all of your points. Resampling would be easy, just go to your library and click **Resample Gestures**. However, performance would be dependent on the device, so there isn't one single solution to this problem.

Problem: Who are you again?
Solution: My name is Oguz Konya. I am a software developer located in Istanbul, TURKEY. I mainly develop games and simulations for customers only.

Problem: How can I contact you?
Solution: Just email me at oguz@oguzkonya.com or oguzkonya34@gmail.com. Please use a clear subject related to product such as "GestureRecognizer can't seem to overthrow the world".