Unity Heat Map Plug-in

Game Engines E2013

Jakob Melnyk, jmel@itu.dk

December 11, 2013

Contents

1 Introduction

2 Project Overview

2.1 Goal of the project

2.2 Plug-in features

3 How To Use

The heat mapping plug-in consists of two parts: tracking objects and generating the heat map.

To access the plug-in, the custom package must be imported. The package consists of 4 folders. *Scripts/HM_Tracker* and *Prefabs/HM_HeatMap* are the elements used by the developer to create heat maps. The rest of the files should normally not be used.

3.1 Tracking game objects

To track a game object, the *HM_Tracker* script must be attached to the object. Attaching the *HM_Tracker* to an object allows the user to change the tracking interval, save interval and which events that are toggled.
The tracker can record the position of the game object when the following events happen: Breadcrumb (object position at intervals), Awake, onDestroy, OnMouseUp and OnTriggerEnter.

INSERT SCREENSHOT OF TRACKER OPTIONS

The OnMouseUp event is only tracked on every call to Update on the game object. Assuming 30 calls to update per second (30 fps), the tracker will register a maximum of 30 mouse up events per second.

The tracking interval determines the minimum interval between every breadcrumb in seconds. This value cannot be less than one second[1].

The save interval is the minimum time between the positions being saved to file[2]. If a game uses a lot of memory, a lower value here would mean less memory is used, but would also mean that the harddrive is accessed more often.

The tracked data is saved in a file named "HeatMapData DATE TIME.xml" in a folder called "HeatMapData". Tracked data for individual game objects are saved with the file name "HeatMap-DataNameOfGameObject.xml" in folder "HeatMapData/DATE TIME/".

Because we track individual game objects in this manner, the names of tracked game objects must be unique. While this issue can be dealt with by the user of the plug-in, this could certainly be an element to improve in a future version of the plug-in.

[1]If the value was lower, it could become a memory hog and take too long to save event info.

[2]Like the tracking interval, this value cannot be less than 1. We felt that saving event information more often than breadcrumbs can be tracked does not make sense.

3.1.1 Custom events

There are two ways to track custom events.

The first way is to create a separate script that calls the AddEvent method of the *HM_Tracker* component on the game object whenever the event happens. This is how we designed custom events to be tracked.

The second way is to modify the *HM_tracker* script, so that the event is tracked from inside the script itself.

3.2 Generating the heat map

A copy of the *Heatmapping/Prefab/HM_HeatMap* prefab should be part of the unity scene hierarchy. The prefab has a script attached that allows for generating heat map visuals. The *Heat Marker* variable of the script should not be changed.

INSERT SCREENSHOT OF HEATMAP OPTIONS

Heat Marker Scale determines the scale of the sphere surrounding a tracked event. *Allowed Distance* determines how far away two positions can be from each other before they count as increased density.

Session Data lets the user choose from different datasets in the "HeatMapData" folder. Currently, it is only possible to load data files from the "HeatMapData" folder.
Once a data file has been loaded, lists of the objects tracked in the file and their events appear in the editor. Selecting a specific object and/or event type will only heat markers for those objects/events in the scene.

Clicking the *Generate Heatmap* button instantiates heat markers based on the selected data file and the game object / event choices. After the objects are instantiated, the density of the objects are calculated and the colours of the heat markers are set based on their density relative to the highest density.

The *Clear Heatmap Markers* button destroys all objects in the same with the tag "HeatMarker".

3.2.1 Loading data from more files

Currently the only way to load data from more than one file is to load (and generate) the heat map data one file at a time. This is highly inconvenient, if there are large data sets (as there often is with heat map data. I discuss possible solutions to this problem in section 6.1 on page 3.

4 Overview of the Program

5 Performance

5.1 Saving data

5.2 Generating the heat map

6 Issues

6.1 Loading data from more files

6.2 Heat map generation performance

6.3 Unique game object names

6.4 Lack of Unity 2D-tool support

7 Conclusion

7.1 Future Work

Binned heatmap

Reducing number of objects in the scene