Other Useful Flash to Unity Functionality

Monday, October 7th, 2013

Abstract:

Flash to Unity is a tool that allows game developers to import animations made in Flash to the Unity 3D Game Engine. This document is intended to give insight on various useful functionality that, while may not be strictly fundamental, is nonetheless useful when creating content with Flash to Unity.

Table of Contents

Introduction

Flash to Unity provides the user with several tools that may assist the user in several tasks that may arise when coding a game, but that on themselves are not components per se. The following sections will talk about the several utilities Flash to Unity offers.

Supplementary Functions

XML parsing

XML parsing is normally used by Flash to Unity to create and manage game settings, but may also be used for other purposes. Two designated classes, F2UXML and F2UTag, are used to hold a representation of the XML tree in memory and decode it accordingly. Their main objective is to encapsulate and abstract C#'s inherent XML parsing functionality, which is normally hard to follow and usually ends up as unreadable for average humans. For more information about the functionality of these classes, please do refer to the reference manual.

Math utilities

Flash to Unity's math library focuses on automating menial chores, specially pertaining vector math. The following subsections will detail the several functions Flash to Unity provides to treat vectors. They suppose basic trigonometric and vector-related concepts like identities and operations over vectors. Only the basic theoretical foundation of each function will be explained; in order to know of the detailed workings, you can consult the reference manual.

Trigonometric functions

Get angle to point

The most basic concept to grasp when simulating movement is getting the angle between a point and a pivot. Given a well-defined coordinate system, it's possible to calculate how many radians a point is located in relation to an arbitrary pivot. For this to happen, the following formula is applied, given a pivot (x, y) and a point (w, v):

$$\theta = atan2(v - y, w - x)$$

Of course, the function lacks any real significance (and is, indeed, undefined) when x = w and y = v. Flash to Unity offers the function GetAngleFromPivot in order to calculate that existing angle.

Rotate point around pivot

Based on the previous concept, one can take a point and a pivot, and rotate it around in a circular fashion a certain amount of units. The utility function GetRotatedPointAroundPivot receives the pivot, the point and the angle to rotate the point to, and returns a new point, which is the previous point rotated by the provided angle. The math to achieve this effect is simple; the provided angle is added to the existing angle between the point and the pivot and then a new position is calculated.

Get a point at a distance

Related to the previous two concepts is the concept to calculate a point at a certain distance from another point, given an angle. This concept is easier to understand if one considers the distance between both points to be the radius of a circle with its center set at the pivot. This way, the location of the point in question becomes easy to calculate; given a radius of r and an angle of Θ, the Flash to Unity function GetPointAtDistance calculates the point (x, y) from pivot (w, v) as follows:

$$x = \cos(\theta) * r + w$$

$$y = \sin(\theta) * r + v$$

Angle conversion functions

It is sometimes convenient to represent angles as degrees instead of radians or vice versa. For this same reason Flash to Unity offers the functions RadiansToDegrees and DegreesToRadians, which convert the desired angle either to degrees or to radians. It is to be noted, though, that every single Flash to Unity function expects angles to be passed as degrees.

Bezier curves and interpolation

Flash to Unity offers functions to calculate points in a Bezier curve as well as interpolating points between several controls, both based on a time step value. This manual won't review the theory on Bezier curves and interpolation in order to avoid verbosity; as for now, it will be enough to mention that Flash to Unity offers utilities to calculate linear, inverse linear, quadratic and cubic Bezier curves (via InverseLinearBezier, LinearBezier, QuadraticBezier and CubicBezier functions, respectively), as well as interpolation through an arbitrary number of points (through the Interp function).