Overview

General demands on Effects

Effects represent the algorithms that transform input data into visible output in the destination framebuffer. Input data can either be images, or data that describes how an image is generated (e.g. Gradient parameters). OpenGL effects are identifiable by the prefix “GL” in their name (e.g. GlLinearGradientBrushBlend). Naturally, these effects are only available on platforms that support OpenGL.

Types of Effects

CGI Studio offers four types of effects. The *Brush Effect* defines its input based on its parameters alone. The *In-place Effect* requires an input (source) and produces an output (destination). The *Blend Effect* requires input (source) and is capable of reading back the target it renders. The *Combined Effect* is a combination of an arbitrary number of other specific effects in a specific sequence in order to have them processed in a more optimized way like a single pass.

Brush Effect

A *Brush Effect* provides means of generating graphical content out of data provided by the user. An example would be an effect which generates a rectangle of solid color based on red, green and blue values provided by an input parameter. A brush effect generates pixel data as output. E.g. SolidColorBrush

In-place Effect

An *In-place Effect* requires source data provided. Source data is always given in form of pixel data and is supposed to be provided by either a Brush Effect or another In-place Effect. This kind of effect can utilize any additional parameters to transform the source pixel data to destination data. The destination data is again pixel data.

Blend Effect

A *Blend Effect* is very similar to an In-place Effect. To be specific, every In-place Effect resembles a specialization of a Blend Effect, namely that they do blend with a fixed functionality (ignoring the target data). What makes Blend Effects special is only their purpose of being the effect which also takes the already computed destination data into account as an input to calculate the new destination data. This allows Blend Effects to transform the source data and the destination data to any kind of combination of both. Destination data is again pixel data. The blend effect is responsible for setting up the alpha blending parameters, how the computed image gets “blended”

into the destination surface (especially the framebuffer).

Combined Effect

A *Combined Effect* is a single effect which represents a whole sequence of other effects chained together. For example, if you want to render bitmap data with a blending function, there usually is a combined effect which represents the Brush Effect and the Blend Effect in one single object. This single object can take the advantage to perform an operation in a single step as the external interfaces (pixel data) can be replaced internally by setting options in the native rendering API. A *Combined Effect* for *Brush* and *Blend Effect* for example is the Candera::BitmapBrushBlend.

Revision #2
Created 2023-02-28 01:22:04 UTC by Tsuyoshi.Kato
Updated 2024-06-11 12:22:36 UTC by Tsuyoshi.Kato