OpenGL ES 3.0 for Apple A7 GPUs and Later

For best performance and to access all of the features of modern GPUs, your app should use Metal. However, if your app is using OpenGL ES, use OpenGL ES 3.0. Using OpenGL ES 3.0 gives you access to new features and a larger pool of rendering resources.

Best Practices

These best practices apply to OpenGL ES 3.0 apps on Apple A7 GPUs and later:

- Avoid operations that modify OpenGL ES objects already in use by the renderer because of previously submitted drawing commands. When you need to modify OpenGL ES resources, schedule those modifications at the beginning or end of a frame. These commands include `glBufferSubData`, `glBufferData`, `glMapBuffer`, `glTexSubImage`, `glCopyTexImage`, `glCopyTexSubImage`, `glReadPixels`, `glBindFramebuffer`, `glFlush`, and `glFinish`.
- Follow the drawing guidelines found in Do Not Sort Rendered Objects Unless Necessary in *OpenGL ES Programming Guide*.

Considerations

The Apple A7 GPUs and later process all floating-point calculations using a scalar processor, even when those values are declared in a vector. Proper use of write masks and careful definitions of your calculations can improve the performance of your shaders. For more information, see Perform Vector Calculations Lazily.

Medium- and low-precision floating-point shader values are computed identically, as 16-bit floating point values. This is a change from the PowerVR SGX hardware, which used 10-bit fixed-point format for low-precision values. If your shaders use low-precision floating point variables and you also support the PowerVR SGX hardware, you must test your shaders on both GPUs.

The Apple A7 GPUs and later do not penalize dependent-texture fetches.

Always use framebuffer discard operations when your framebuffer contents are no longer needed. The penalty for not doing so is higher than it was on earlier GPUs. For best results, use the `GLKView` class; it automatically implements framebuffer discard operations.

When rendering to multiple targets, limit your app to four image targets (and no more than 128 bits of total data on Apple A7 GPUs and 256 bits of total data on Apple A8 GPUs written to the targets). A single `sRGB` target counts as 64 bits.

Copyright © 2018 Apple Inc. All Rights Reserved. Terms of Use | Privacy Policy | Updated: 2018-06-04