Unity performance optimization techniques

Source: <https://cgcookie.com/articles/maximizing-your-unity-games-performance>

* **Keep It Simple**

I'm putting this first since it should be a general rule for any game that you build. Whenever you design a game you need to determine specifically what you need and, more importantly, what you don't need to include.

I enjoy going back to old scripts or projects and finding ways to make something more efficient, often by stripping down the excess. However, think with performance in mind as you're designing it the first time. Don't limit yourself too much, but understand that it's easier to build a performant game from the start rather than trying to restructure things to work better later.

* **Use the Profiler**

Before you start removing lines of code, refining prefabs, and trying to make everything performant, you need to know what is actually causing performance issues. The Profiler is a great way to get an in-depth look at how your game is performing.

* **Batch Those Game Objects**

Often, the visual aspects of a game are going to be one of the big areas in which you can improve performance. Visual elements will impact draw calls. To put it simply, anything that appears on screen must be "drawn". Imagine having 100 different draw calls for a scene versus optimizing your scene to have less than 5.

* Reduce and Reuse Textures

Since batching works based on like materials, you can combine many objects together if they share one big texture. Multiple high-resolution textures will slow down performance. While you can have these in your game, you need to ensure that you're being selective about how they are being used.

* Use Culling to Limit What is Rendered

This is one of my favorite ways to increase performance, but maybe just because I like the way it looks when testing.

* Optimize Objects that are Visible

Limiting texture sizes and combining meshes are a great way to improve performance, along with culling of all sorts. How can we improve the performance of objects that are not being culled but are too far to see in detail? LODs (Levels Of Detail) are a way to render a lower poly version of a mesh when it's outside of a certain range. A mesh that is quite far away can be made incredibly low poly to improve performance.