Scalability and The Developer

An overview of Scalability options and considerations for content creators, testers, programmers, and managers.

Scalability is more than just graphics settings. It is about choosing a target platform and making decisions about art, gameplay, sound, AI, and any number of other systems to fit that platform. For consoles, the platform is fixed, and this is more straightforward. However, for PC, the number of different combinations makes it impossible to focus on a single hardware specification.

This is where the Scalability options come into play. Even before they need tweaking, the individual developer can do a lot to make sure the application they are producing will run on their chosen minimum-specifications hardware.

General

Different machines can have different performance, memory, and quality characteristics, such as:

- Hard drive size, latency, and bandwidth.
- Main memory size, latency, and bandwidth. Running 32-bit instead of 64-bit can limit this further.

- CPU (one or more main processors, hyper-threading, SSE features, branch prediction).
- GPU (memory, Multi GPU, bandwidth, features, shared memory).
- Resolution (modern notebooks have high-resolution displays but slow GPUs).

Graphics is the easiest to scale without affecting gameplay much. Players can adjust scalability settings from your application's options menu. These give the player control over a lot of decisions that are subjective and have tradeoffs:

- Frames per second
- Resolution
- Image quality
- Motion Blur
- Texture details
- Flickering (anti-aliasing)
- Battery life

Artists and Designers

Performance and memory requirements are directly connected to content count and quality. This is a very complex problem:

- Content is often created without the target hardware.
- A complex matrix of hardware properties (low memory but fast, fast GPU but slow CPU).
- The engine code is still under development and optimized on needs driven by content.
- New hardware and drivers change the equation.
- Development time constraints limit optimization efforts.

In order to make this manageable, it is best to have some good practices:

- Plan and verify budgets (triangle / object / material / light count, texture / mesh / sound memory, etc.).
- Add scalability options to non-gameplay relevant content, test with lower and higher settings.
    - e.g. hide clutter objects on lower specs, use simpler material shading, object level of detail.
- Restrict player abilities.
    - e.g. visibility blocker, cannot build arbitrary complex structures, limit destruction.

- Understand performance characteristics.
  - e.g. many dynamic objects with many point lights, mesh particles vs. sprite particles.
- Mix different components in similar ratios (some hardware might be only slow for some features)
  - e.g.: avoid having many objects simple material, few objects many lights with complex materials.
- Understand in-engine profiling tools.

QA

A wide range of scalability costs options more QA time. A few good practices can help:

- Document low, recommended, and high specs.
- Test on a wide matrix (low / med / high GPU, low / high CPU, low / high memory).
- Define performance levels on reproducible scenes.
- Document and communicate changes over time.
- Disabling things is a powerful way to isolate a performance issue (e.g. show console command).
- If you run in lowest quality settings and something looks quite better than expected, it might miss a scalability setting (fixing this could mean more players can enjoy the game).
- Test usability of the scalability: report anything if you find it confusing and wrong.
- At any time, the user interface should be readable and functional. Test low-resolution and lowest-quality settings, and look for unreadable fonts and elements outside of the screen.
- If some artifacts only happen at some specific scalability level (e.g. only on "low") you can isolate the issue further (e.g. shadows are likely affected by "shadow quality") going down to the actual console variable setting (look into the `Scalability.ini` file).
- Know how to quantify performance (stat FPS, stat unit, ...). Milliseconds are more useful than FPS ("it became 10FPS faster" is not meaningful, but "it became 5ms faster" is).
- Some notebooks have multiple GPUs. By default, you want to run on the faster GPU but it is best to run test both (use the driver setting to change that). When running on battery, it might pick the slower one or even switch it at runtime.

Programmers

A player might have an advantage seeing the shadow of a player earlier, or when having a higher frame rate gives them a faster response time. Staying within the offered scalability settings is not cheating, but some scalability settings could be abused to cheat. A console variable might offer a wider range of values than needed for the scalability, or a combination of console variable settings can lead to problems. Some multiplayer games address that by giving control to the server administrator, overriding problematic setting to offer all players equal chances.

For example, hiding in deep grass can be a good gameplay tactic. However, if grass is no longer rendered in larger view distances, this can feel very unfair to the player trying to hide.

Ideally, no scalability setting should affect gameplay.

Leads, Producers, and Managers

Developing a game that needs to scale to a wide range of hardware is harder and costs more time and QA efforts. A few good practices could help:

- Define a low spec and a recommended spec early in the project development.
- Set good defaults, as most players never use the options menu. To get the defaults right, you need a lot of iterations and testing.
- A smaller scalability range makes development easier, as opposed to a broad range (e.g. high-end PC to low0end mobile).
- Having a second low-spec machine can help in every discipline (code, design, and art). When balancing tasks, it is generally better to optimize than to create more scalability.
- Performance is one part of scalability, but it can also affect stability. For example, If you run out of memory on a lower spec, it might be best to grey out the high-resolution texture option.
- Scalability on low-end devices means more players can play the game. Scalability on high-end devices means the game can look better, especially in screenshots and tests.