One-Pager: Media Management

Common Uses of Media in Games Similar to Mine

Games in the top-down space-survival genre rely heavily on media assets to communicate gameplay clarity and mood. Common uses include visual effects for movement (thruster flames, shield glow), distinct obstacle and enemy sprites, collectible icons that are readable at a glance, and UI elements that clearly represent health, cargo, or upgrades. Sound effects are also important for feedback—collecting items, shooting, taking damage. Background music or ambient space sounds help create immersion and give the environment a sense of motion even when the player is standing still.

Strategies for Integrating Media (Pros & Cons)

1. Direct Asset Embedding (Sprites, Audio, UI imported normally)
Pros:

- Very easy to set up in Unity.
- Quick iteration cycle.
- Works well for small or medium-sized projects.
- No additional programming or tools required.

Cons:

- Can get unorganized quickly if many assets are added.
- Harder to manage alternate versions of assets (e.g., different resolutions).
- May increase loading time if too many large files are included.

2. Asset Bundles or Addressables
Pros:

- Great for organization and scalability.
- Supports loading/unloading assets at runtime, saving memory.

- Allows updating or swapping assets without rebuilding the whole game.

Cons:

- More complex to set up and maintain.
- Overkill for small student projects.
- Requires extra debugging when referencing assets asynchronously.

Approach I Will Use and Why

For my project, I will use **direct sprite/audio embedding** because my game is small in scope, and this method allows me to stay focused on gameplay rather than asset pipelines. The media I need—rock/asteroid sprites, gear collectibles, UI icons, portal effects, and simple sound cues—are lightweight and don't require runtime swapping or version control complexity. I'll keep everything organized by separating folders into categories such as Sprites/Obstacles, Sprites/Components, Audio/SFX, and UI. This approach gives me the fastest iteration speed, reduces technical overhead, and ensures stable performance while still supporting the visual and audio clarity the game needs.