PROJECT COLOMBO:

NAMING CONVENTIONS & WORKFLOW

CONTENT

NAMING CONVENTIONS OVERVIEW

Having clear and consistent naming conventions across all departments is critical to maintaining organization, efficiency, and readability throughout the project. This applies to art assets, code, sound files, and project management documents. Well-structured files will ensure that anyone in the team can easily understand and locate assets without confusion, streamlining the workflow.

FILE NAMING CONVENTIONS

1. ***Clear and Descriptive Names:***

All file names must describe the contents of the file. Use a maximum of 3 words to keep names concise but informative. Example:

`CharacterConcept` , `EnemyModel_Arlecchino`

2. ***Iterative Naming:***

When multiple iterations of a file exist, append a version number after an underscore.
Example:

`CharacterConcept_1` , `CharacterConcept_1`

FILE NAMING CONVENTIONS

3. ***File Types:***

For sound files and visual effects, begin the name with a prefix indicating the file type.
Example:

SFX_MetalHit , OST_MainTheme , VFX_Explosion

4. ***Folder Structure:***

All departments should categorize files into clear, logical folder structures. For example::

- *Art*: Concepts, UI, 3D Models, Textures, Materials
- *Audio*: OST, SFX
- *Programming*: Scripts and respective sub-folders

FILE NAMING CONVENTIONS

5. ***Handling Deprecated Files:***

Place files that are no longer in use in a folder named "Deprecated" to keep the workspace clean.

PROGRAMMING NAMING CONVENTIONS

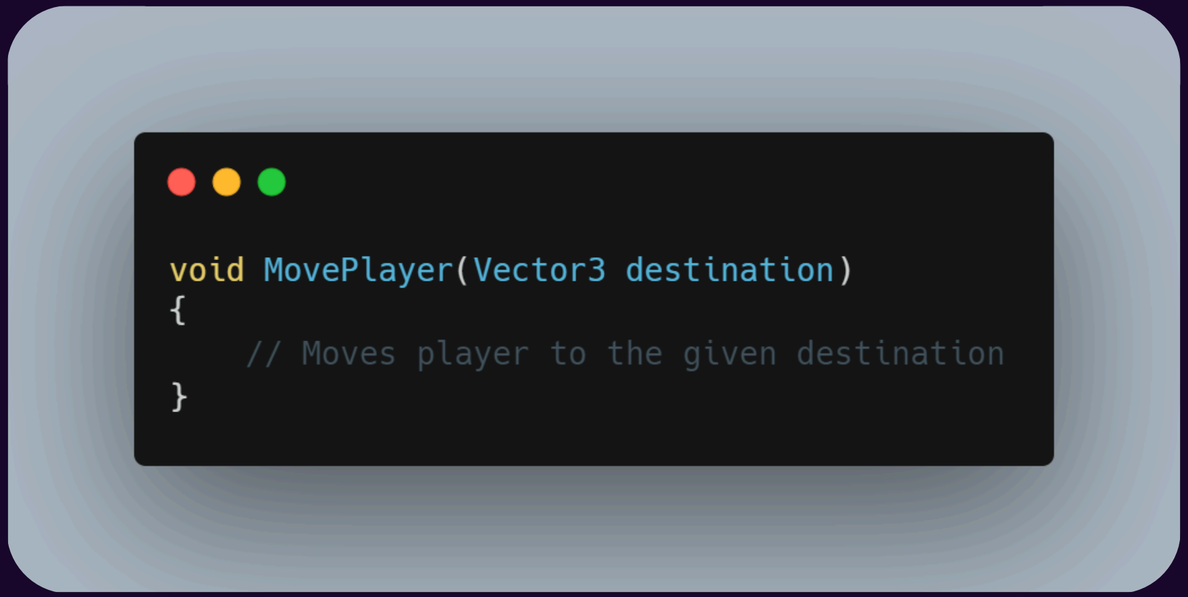
1. ***Variables:***

Use **camelCase** for variable names and always choose descriptive, non-abbreviated names.

PROGRAMMING NAMING CONVENTIONS

2. ***Functions:***

Use **PascalCase** for function names, starting each word with a capital letter. Ensure function names are clear and indicate their purpose.



PROGRAMMING NAMING CONVENTIONS

3. ***Bracket Style:***

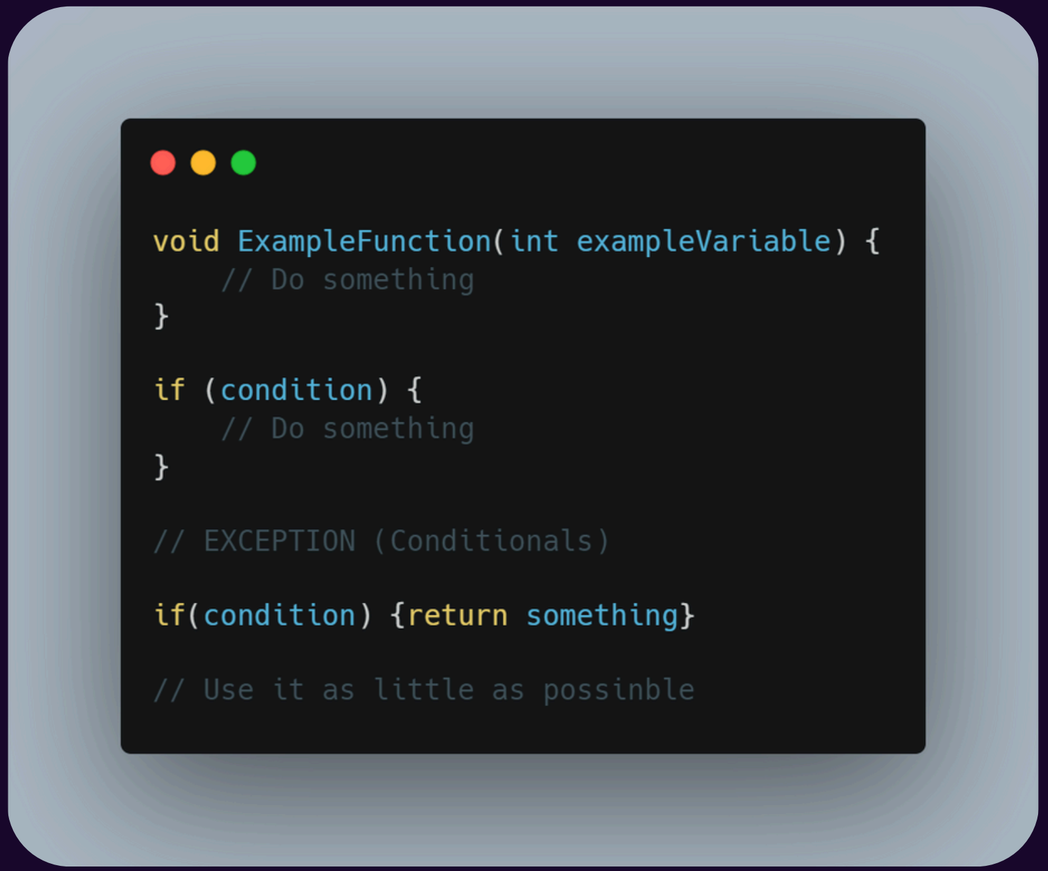
Use the following bracket style for all code structures:

```
void ExampleFunction(int exampleVariable)
{
    // Do something
}

if (condition)
{
    // Do something
}

//etc etc (loops, and other code structures)
```

PROGRAMMING NAMING CONVENTIONS

3. ***Bracket Style THAT I NEVER WANT TO SEE:***

Everytime you use this, a programmer dies:



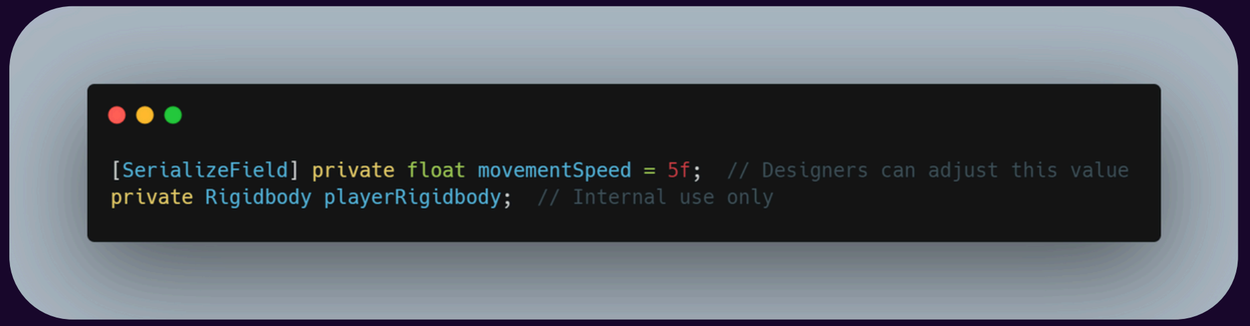
EXPANDED PROGRAMMING STANDARDS

1. ***Consistency:***

Always maintain consistency in your naming conventions, brackets, and code formatting. This increases the readability and maintainability of the codebase

2. ***[SerializedField] variables:***

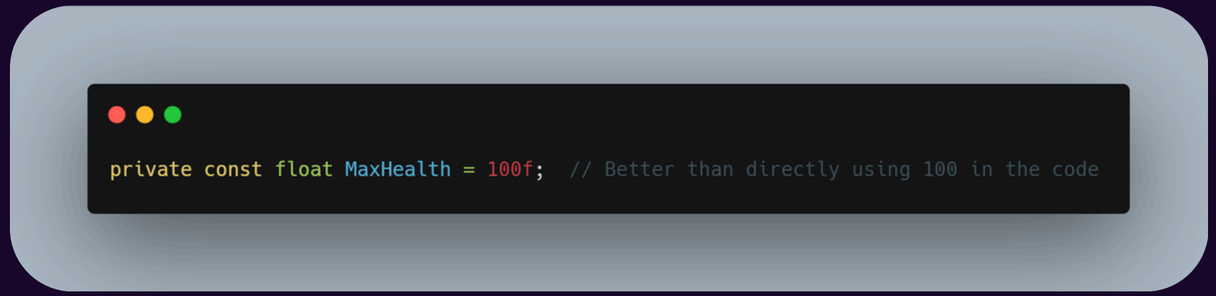
Only expose variables to the Unity Inspector if needed by designers or if they are configurable. All essential variables should remain private to avoid unnecessary changes. Example:



EXPANDED PROGRAMMING STANDARDS

3. ***Avoid Magic Numbers:***

Never hard-code values directly into logic. Use variables to make your code adaptable and easy to adjust.
Example:



4. ***Commenting and Documentation:***

Comment your code to explain complex logic, but avoid excessive comments for simple and obvious code. It doesn't need to be many comments or very long, just to understand what the code does generally!

UNITY WORKFLOW OVERVIEW

As we transition into **pre-production** in **Unity**, our primary focus will be establishing **core mechanics** and f**oundational systems**. We will streamline the workflow to ensure smooth integration across all departments.

UNITY WORKFLOW STEPS

(October – December 2024)

1. ***Assign Roles & Define Tasks:***

Each team member should be assigned tasks based on their expertise. Tasks should focus on building core systems in Unity (e.g., player movement, AI, level design).

2. ***Core Mechanics Development:***

The programming team will focus on core gameplay mechanics such as player movement, combat, and interactions. Prototypes should be created and tested for functionality.

3. ***Art and Asset Creation:***

While core mechanics are developed, the art team will work on concept designs, models, and animations. No assets will be implemented in Unity yet until a stable build is ready for integration.

4. ***Version Control:***

Ensure all assets and scripts are uploaded to our version control system (GitHub) for collaborative work. No individual should integrate changes directly without approval from the project leads.

UNITY WORKFLOW STEPS

(October – December 2024)

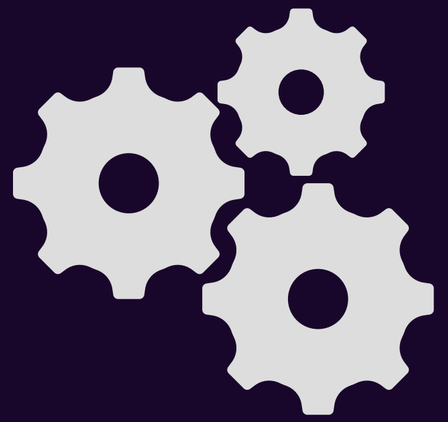
5. ***Designers & Artists Collaboration:***

Designers and artists should review the functionality regularly to ensure everything meets the game's design vision.

PRE-PRODUCTION & TASKS OVERVIEW

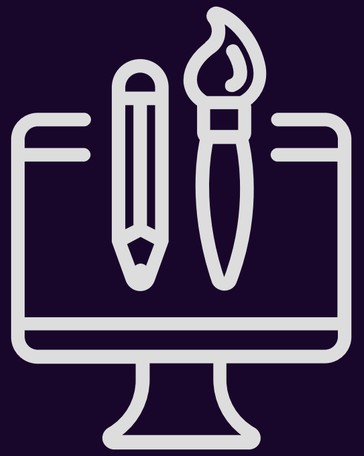
1. ***Core Mechanics in Unity:***

By November, our focus will be building a basic prototype of core mechanics like movement, AI, and combat. This will serve as the foundation for asset integration later.



2. ***Asset Preparation:***

The art team will continue preparing assets, but nothing should be implemented in Unity without approval from leads. This prevents clutter and keeps the project clean.



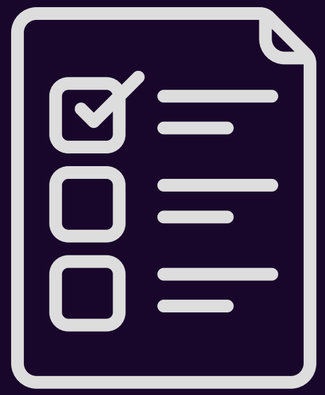
PRE-PRODUCTION & TASKS OVERVIEW

3. ***GDD Updates:***

Game designers should ensure the Game Design Document is updated with finalized core features and gameplay mechanics.

4. ***Task Assignment & Deadlines:***

Task assignments should follow a clear schedule with deadlines. Utilize project management tools to track progress and communicate with the team.



PROJECT MANAGEMENT & ROLES

1. ***Team Assignments:***

- **Programmers**: Focus on core systems and mechanics.
- **Artists**: Continue working on assets (concepts, models, animations) for the final production phase.
- **Designers**: Test core systems and ensure they align with the game’s vision.

PROJECT MANAGEMENT & ROLES

2. ***Communication & Approvals:***

- No assets or mechanics should be added to Unity without approval from project leads.
- Weekly meetings to review progress and adjust schedules or tasks as needed.

3. ***Task Management Tools:***

- We use project management software like Miro and Jira to assign, track, and review tasks.

CONCLUSION

By following these naming conventions and workflow processes, we will maintain clarity and efficiency across all departments, ensuring that we meet our pre-production goals by December. Let’s continue to focus on laying the right foundations for the full production phase starting next year.