* **GAME APPLICATION LAYER (Bộ thư viện của Unity đã xử lý)**
* Extension:
* Xử lý XMLSerialize
* Tương tác với SQLite
* **GAME LOGIC LAYER:**
* GameManager:
* Quản lý trạng thái của game (Pause, Resume, Game over, Game succeed, Enter, Exit).
* Khởi tạo các thành phần trong game (Player, MapManager, QuestManager, DataManager, EventManager, EnemyManager, UIManager, SoundManager)
* Gửi yêu cầu Load/Save game.
* EventManager:
* Là đơn vị truyền thông điệp giữa các đối tượng trong game
* Có chức năng nhận đăng ký sự kiện từ các đối tượng và thông báo cho chúng khi sự kiện xảy ra
* Có chức năng xóa sự kiện và thanh lọc sự kiện (xóa những sự kiện mà đối tượng nhận trở thành null)
* MapManager:
* Quản lý việc load và khởi tạo map
* Gửi yêu cầu đến các đối tượng liên quan khi nhân vật đi qua những trigger trên map
* TriggerZone:
* Kiểm tra va chạm tại những trigger trên map
* Gửi sự kiện khi trigger đó xảy ra
* QuestManager: (??????????????????)
* DataManager:
* Xử lý Load/Save Game
* Xử lý load các tài nguyên trong game
* Xử lý load thông tin các đối tượng trong game (Chỉ số nhân vật, quái vật). Có thể sử dụng XML hoặc SQLite
* PlayerController:
* Quản lý trạng thái của nhân vật
* Tính toán và xử lý va chạm rồi gửi yêu cầu cập nhật các thành phần khác như PlayerAnim, UIManager,…
* EnemyManager:
* Khởi tạo Enemy theo kịch bản
* Điều khiển trạng thái của tất cả enemy (Group all)
* EnemyController:
* Quản lý trạng thái của Enemy
* Tính toán và xử lý va chạm rồi gửi yêu cầu cập nhật các thành phần khác như EnemyAnim, EnemyAI,…
* CharacterAbility:
* Quản lý các chỉ số của những nhân vật. Cập nhật thường xuyên
* DamageDealer:
* Phát hiện và xử lý khi nhân vật va chạm với vật thể gây damage
* **GAME VIEW LAYER:**
* **Game View: Human**
* UIManager:
* Cập nhật hệ thống UI trong quá trình game diễn ra.
* Xử lý khi người dùng tương tác với các control UI.
* SoundManager:
* Thực thi play audio
* Cache tài nguyên âm thanh đã được load
* PlayerAnim:
* Cập nhật animation sao cho phù hợp với trạng thái của nhân vật trong PlayerController.
* PlayerInput:
* Vòng lặp kiểm tra input điều khiển nhân vật từ người dùng
* GameSettings:
* Quản lý 1 vài settings cho người dùng như âm thanh
* Load/Save settings của người dùng thông qua XML
* **Game View: AI**
* EnemyAI:
* Kiểm tra những hoàn cảnh cụ thể của game (Stimulus Interpreter)
* Trí thông minh của enemy (Decision Making)
* Tự động gửi yêu cầu thay đổi trạng thái tới EnemyController dựa theo hoàn cảnh cụ thể trong game
* EnemyAnim:
* Cập nhật animation sao cho phù hợp với trạng thái của nhân vật trong EnemyController