

# SAE-TeMV1-2324 - Cube Game Audio Document

## Cross Character

- Event: **Jump**
  - Evento lanciato quando uno dei character salta
  - Considerare che questo evento è unico per tutti i characters del gioco, quindi la caratterizzazione di un suono specifico per il player va ad influire anche sugli altri characters

## Player

- Event: **Foot\_Player** - Switch: **Material (Concrete, Grass, Gravel, Metal, Sand, Stone, Tile, Water, Wood)**
  - Creare un sistema in grado di cambiare suono in base alla superficie dove si trova il Player.
  - La percezione di ripetizione nella riproduzione dei footsteps deve essere evitata
  - Creare con la tecnica del Layering un sistema che non comprenda solo il suono dei footsteps ma anche altri asset che diano un senso di movimento realistico
  - Prendere in considerazione i momenti in cui i passi non si devono sentire
  - La frequenza dei passi deve essere realistica a prescindere da quante volte viene mandato l'evento dal gioco
- Event: **Pain**
  - Questo evento viene lanciato quando il player viene colpito
  - Creare un sistema che evidenzi questo evento di gioco
- Event: **Spawn\_Player**
  - Viene Lanciato quando il player viene creato in game
  - Creare un asset che evidenzi questo evento di gioco
- Event: **Death\_Player**
  - Viene Lanciato quando il player muore
  - Questo evento presenta dei bug, implementare comunque un suono che evidenzi questa azione
- Event: **SplashIn\_Player / SplashOut\_Player**
  - Vengono lanciati rispettivamente quando il player entra ed esce dall'acqua
  - Oltre a emettere un suono adatto all'azione questo evento può essere usato per regolare altre condizioni del mix e sound design
- Event: **Land**
  - Evento lanciato quando si cade da una grande altezza dopo un salto
  - Creare la massima variazione per questo evento

# Player - Weapons & Objects

- Event: **WeapLoad**
  - Questo evento viene lanciato quando si cambia arma
  - Creare un suono di UI e Fisico
- Event: **ItemPup**
  - È un potenziamento delle abilità del player chiamato Quad nel gioco che influenza la potenza di fuoco delle armi
  - Viene lanciato quando si raccoglie l'oggetto
  - Il Sound design del gioco deve evidenziare la differenza di condizione tra lo stato di Quad e lo stato normale
- Event: **PupFire**
  - Evento che viene lanciato quando si spara sotto l'effetto del Quad
  - Viene lanciato ogni volta che si spara
  - Usare questo evento per sottolineare la potenza aumentata dell'arma
- Event: **PupOut**
  - Evento lanciato quando finisce l'effetto del Quad
  - Implementare un evento sonoro che evidensi questo cambio di condizione
- Event: **NoAmmo**
  - Evento lanciato quando non ci sono più munizioni disponibili
- Event: **ItemAmmo**
  - Evento lanciato quando si raccolgono le munizioni di un'arma
- Event: **ItemArmour100 / ItemArmour150**
  - Evento lanciato quando si raccoglie l'armatura
- Event: **ItemHealth100 / ItemHealth200**
  - Evento lanciato quando si raccolgono i medkit o le pills bottle
- Event: **Fire\_Chaingun\_Player**
  - Evento lanciato quando si spara con la mitraglietta
- Event: **Fire\_Rifle\_Player**
  - Evento lanciato quando si spara con il fucile
- Eventi **Fire\_Shotgun\_Player**
  - Evento lanciato quando si spara con il fucile a pompa
- Event: **Fire\_Rocket\_Player / Hit\_Rocket\_Player**
  - Eventi Lanciati rispettivamente quando viene lanciato il missile e quando il missile esplode
- Event: **Fire\_Fist\_Player**
  - Evento lanciato quando il player tira un pugno



# Environment

- Event: ***Map\_Loaded***
  - Questo evento viene lanciato ogni volta che viene caricata una mappa
  - Utilizzare questo evento per gestire i parametri ed assicurarsi che tutto nel gioco funzioni correttamente
- Aggiungere un evento per la gestione degli ambienti per ogni mappa
  - **metl3**
  - **mak2**
  - **dusk**
  - **hellsgate**
- Event: ***Emitter\_Teleport***
  - Evento collegato al Teleport rotante, viene lanciato al caricamento della mappa
- Event: ***Teleport***
  - Evento lanciato quando il giocatore passa attraverso l'oggetto di teletrasporto
- Event: ***ItemSpawn***
  - Evento lanciato quando un oggetto (munizioni, armatura) ricompare sulla mappa
- Event: ***JumpPad***
  - Evento lanciato quando si salta sulle piattaforme che lanciano i character in aria

# Enemies

		
Bauul	Goblin	Knight
		
HellPig	Ogre	Rat
		
Slith	Rhino	

- Event: ***Foot\_ "NameOfMonster"*** (ex: Foot\_Ogre) Switch: ***Material***
  - Creare un sistema per i footstep dei Nemici
  - Si possono riutilizzare i footstep per classi di nemici comuni, ad esempio HellPig e Ogre. Fare almeno due distinzioni tra i nemici.
  - In scene dove ci sono tanti nemici non sovraccaricare il mix di suoni di footsteps
  - Creare con la tecnica del Layering un sistema che non comprenda solo il suono dei footsteps, ma anche altri asset che diano un senso di movimento realistico considerando anche il tipo di nemico.
- Event: ***SplashIn\_Monster / SplashOut\_Monster***
  - Evento lanciato quando i nemici entrano ed escono dall'acqua
- Event: ***Spawn\_Monster***
  - Evento lanciato quando i nemici compaiono nella mappa
- Event: ***Grunt\_ "NameOfMonster"*** (ex: ***Grunt\_Ogre***)
  - Evento lanciato quando il nemico si accorge del giocatore, viene lanciato molto raramente ma implementare comunque testando in Wwise
- Event: ***Pain\_ "NameOfMonster"*** (ex: ***Pain\_Ogre***)
  - Evento lanciato quando i nemici vengono colpiti
- Event: ***Death\_ "NameOfMonster"*** (ex: ***Death\_Ogre***)
  - Evento lanciato quando i nemici vengono uccisi
- Event: ***Fire\_Bite\_Monster***
  - Evento lanciato quando Hellpig morde
- Event: ***Fire\_Chaiungun\_Monster***
  - Evento lanciato quando Rhino Spara
- Event: ***Fire\_Fireball\_Monster / Hit\_Fireball\_Monster***
  - Evento connesso alla Fireball lanciata da Ogre, i due eventi vengono lanciati rispettivamente al lancio ed all'impatto
- Event: ***Fire\_Icebball\_Monster / Hit\_Icebball\_Monster***
  - Evento connesso alla Iceball lanciata da Knight, i due eventi vengono lanciati rispettivamente al lancio ed all'impatto
- Event: ***Fire\_Slimeball\_Monster / Hit\_Slimeball\_Monster***
  - Evento connesso alla SlimeBall lanciata da Goblin, i due eventi vengono lanciati rispettivamente al lancio ed all'impatto
- Event: ***Fire\_Shotgun\_Monster***
  - Lanciato da Rat
- Event: ***Fire\_Rifle\_Monster***
  - Lanciata da Slith
- Event: ***Fire\_Rocket\_Monster / Hit\_Rocket\_Monster***
  - Evento collegato a quando i mostri lanciano un missile
  - Si possono utilizzare gli stessi assets usati per il player

## Music

- Per la musica interattiva usare l'evento **Map\_Loaded**
- Event: **Music\_Off**
  - Implementare un evento che silenzia la musica.
- Creare un sistema per la musica interattiva con questi requisiti:
  - Usare l'Interactive Music Hierarchy per l'implementazione della musica
  - Il sistema di musica interattivo deve reagire ad almeno un evento in game (esempio: Arrivo dei Nemici)
  - Il sistema deve implementare randomizzazione e sequenzialità sia verticalmente che orizzontalmente.
  - Per questo sistema possono essere usati gli assets forniti dal docente.

## Consegna & Ottimizzazione

- Le soundbanks devono essere generate per due piattaforme (Windows, Mac)
- I suoni appropriati devono essere mandati in stream direttamente dal disco
- Settaggi minimi di compressione
  - Mac: Sample Rate: 48k / Format: Vorbis / Quality: 4
  - Windows: Sample Rate: 48k / Format: Vorbis / Quality: 2
- Rendere Mono i file che non ha senso siano Stereo (es. Shotgun Shells)
- Nella Cartella devono essere inclusi i seguenti elementi:
  - Progetto Wwise
  - Soundbanks per Windows e Mac
  - Foglio con le note di Lavoro Generali
  - Tabella Excel con lista degli eventi e note di lavoro
  - Cartella del gioco Cube con le mappe personalizzate. All'interno della cartella cube devono essere presenti le seguenti cartelle e file
    - cube
    - cube\_source
    - Cube.png
    - Mac
    - sample.json
  - se previsto, la sessione di lavoro della DAW utilizzata per la creazione o registrazione dei Foley.