GENERATION DE BUTIN ALEATOIRE

Le but est de déterminer quel butin est généré par un ennemi tué, en fonction de sa puissance et de la rareté de l’objet que l’on veut voir.

# CLASSE DE LOOT

De la même manière que les ennemis ont des classes qui définissent leur puissance, on définit des classes de loot :

* **Alpha** : souvent aucun butin, parfois de l’argent, rarement de l’équipment.
* **Beta** : parfois aucun butin, souvent de l’argent, et parfois un objet d’équipement.
* **Delta** : rarement aucun butin, souvent de l’argent ou un objet d’équipement.
* **Gamma** : toujours un objet d’équipement : souvent de qualité, parfois épique.
* **Unique**: toujours un objet épique.
* **Zeta**: aucun butin
* **Shop**: toujours un objet d’équipement (cette classe permet de générer des magasins).

# VALEUR DE LOOT

La valeur du butin est proportionnelle à celle de l’ennemi, ou du magasin. Chaque ennemi ou magasin se voit donc configurer une valeur entière : « **value** ».

# BEAST SCALE

La rareté du butin et donc le nombre et la puissance de ses caractéristiques pour une valeur de butin donnée, dépend d’un facteur baptisé « **Beast Score** ».

Pour introduire une part d’aléatoire dans les résultats, le Beast Score de l’ennemi est à chaque fois tiré à l’intérieur d’une fourchette de valeurs min et max appelée Beast Scale, définie pour chaque classe d’ennemi.

Le Beast Score du butin a 3 composantes que l’on additionne :

* Beast Score de l’ennemi : valeurs min et max définies pour chaque classe d’ennemi.
  + Alpha : {0.7 / 1.3}
  + Beta : {0.9 / 1.5}
  + Delta : {1.1 / 1.7}
  + Gamma : {1.5 / 2.2}
  + Unique : {6 ; 7}
* Beast Score du joueur : dépend de la combo du joueur.
  + Red combo : +0.2
  + Blue combo : +0.4
  + White combo : +0.6
* Beast Score du niveau : dépend du HazardLevel du niveau.
  + +0.5 par Hazard Level

# GENERATION DU LOOT

* On fait un tirage aléatoire pour déterminer si l’ennemi a généré un butin ou pas. Cette probabilité dépend directement de la classe d’ennemi : **LootTable\_Dropping\_Something[ ].**
* Si il y a un butin, on fait un second tirage pour savoir s’il s’agit d’un objet d’équipment. Cette probabilité dépend aussi directement de la classe d’ennemi : **LootTable\_Dropping\_Equipment[ ].**
* Si ce n’est pas un équipment, on génère un butin d’argent.

# GENERATION DU BUTIN D’ARGENT

* Argent = **value** \* **Beast** **Score**
* Le butin est créé :

D:\GitHub\StarFighter\Unstable\StarFighter\StarFighter\Assets\2D\crystal.png

# GENERATION DU BUTIN D’EQUIPMENT

* On détermine des fourchettes de valeurs Beast Score à l’intérieur desquelles on garantit un nombre de propriétés bonus pour l’équipement, plus une propriété supplémentaire qui a une chance d’être ajoutée : plus le Beast Score est près du seuil supérieur, plus la propriété supplémentaire a de chances d’être ajoiutée.
  + Beast Score compris entre 0 et 1.5 : 0 propriété garanties, chance d’1 propriété.
  + Beast Score compris entre 1.5 et 3 : 1 propriété garanties, chance de 2 propriétés.
  + Beast Score compris entre 3 et 4.5 : 2 propriété garanties, chance de 3 propriétés.
  + Beast Score compris entre 4.5 et 6 : 3 propriété garanties, chance d’épique

Note : en effet, si l’équipement a 4 propriétés garanties, on donne un objet éqique.

Les objets épiques ne sont pas générés procéduralement, mais pré-définis.

* On détermine le type d’équipment par un simple tirage aléatoire parmi tous les types d’équipement existants + le type Arme.
* On génère les caractéristiques de base de l’objet, en fonction logarythmique de sa **value**.
* On fait un tirage aléatoire sans remise parmi les propriétés possibles pour ce type d’équipement jusqu’à avoir satisfait le nombre de propriétés défini pour ce butin, à moins qu’il n’existe pas suffisamment de propriétés uniques pour ce type d’équipement (dans ce cas-là, le surplus est perdu).

Chaque propriété ajoute à une ou plusieurs caractéristiques de l’objet, en fonction logarythmique de sa **value** et de son **Beast** **Score**.

* L’objet est créé :

 

ANNEXE

Fonctions logarythmiques entre les bornes 100 et 1000.

