**CubeIt – generování objektu Reward**

# Raw code (Kotlin)

@Exclude fun generate(inPlayer: Player = Data.player): Reward {  
 if (this.type == null) {  
 this.type = when (nextInt(0, GenericDB.balance.itemQualityPerc["7"]!!+1)) { //quality of an item by percentage  
 in 0 *until* 3903 -> 0 //39,03%  
 in (GenericDB.balance.itemQualityPerc["0"] ?: 1) + 1 *until* (GenericDB.balance.itemQualityPerc["1"] ?: 1) -> 1 //27%  
 in (GenericDB.balance.itemQualityPerc["1"] ?: 1) + 1 *until* (GenericDB.balance.itemQualityPerc["2"] ?: 1) -> 2 //20%  
 in (GenericDB.balance.itemQualityPerc["2"] ?: 1) + 1 *until* (GenericDB.balance.itemQualityPerc["3"] ?: 1) -> 3 //8,41%  
 in (GenericDB.balance.itemQualityPerc["3"] ?: 1) + 1 *until* (GenericDB.balance.itemQualityPerc["4"] ?: 1) -> 4 //5%  
 in (GenericDB.balance.itemQualityPerc["4"] ?: 1) + 1 *until* (GenericDB.balance.itemQualityPerc["5"] ?: 1) -> 5 //0,5%  
 in (GenericDB.balance.itemQualityPerc["5"] ?: 1) + 1 *until* (GenericDB.balance.itemQualityPerc["6"] ?: 1) -> 6 //0,08%  
 in (GenericDB.balance.itemQualityPerc["6"] ?: 1) + 1 *until* (GenericDB.balance.itemQualityPerc["7"] ?: 1) -> 7 //0,01%  
 else -> 0  
 }  
 }else {  
 this.type = kotlin.math.*min*(7, this.type!!)  
 }  
 when (this.type) {  
 0 -> {  
 when (nextInt(0, 101)) {  
 in 0 *until* 75 -> this.type = 0  
 in 76 *until* 100 -> this.type = 1  
 }  
 }  
 7 -> {  
 when (nextInt(0, 101)) {  
 in 0 *until* 75 -> this.type = 7  
 in 76 *until* 100 -> this.type = 6  
 }  
 }  
 else -> {  
 when (nextInt(0, 101)) {  
 in 0 *until* 10 -> this.type = this.type!! + 1  
 in 11 *until* 36 -> this.type = this.type!! - 1  
 }  
 }  
 }  
 if (nextInt(0, 100) <= ((this.type!! + 1) \* 10)) {  
 item = GameFlow.generateItem(inPlayer, this.type)  
 }

cubeCoins = nextInt((GenericDB.balance.rewardCoinsBottom \* (inPlayer.level \* 0.8) \* (this.type!! + 1) \* 0.75).toInt(), GenericDB.balance.rewardCoinsTop \* ((inPlayer.level \* 0.8) \* (this.type!! + 1) \* 1.25).toInt())  
 experience = nextInt((GenericDB.balance.rewardXpBottom \* (inPlayer.level \* 0.8) \* (this.type!! + 1) \* 0.75).toInt(), (GenericDB.balance.rewardXpTop \* (inPlayer.level \* 0.8) \* (this.type!! + 1) \* 1.25).toInt())  
 if(this.boss) gold = nextInt(10 \* this.type!!)  
  
  
 if(item != null) this.decreaseBy(1.75, false)  
  
 return this  
}

# Logický postup generování

**Generování „kvality“** – často děděná kvalita od obtížnosti např. misí – obtížnost 7 = vstupní kvalita 7. V případě, že není vstupní kvalita, kvalita se generuje dle pravidel pro generování (balance – GenericDB/Balance).

**Úprava kvality –** je potřeba generování kvalit založit více na náhodě. V případech, že jsou kvality krajové (tedy limity – 0 a 7), mají 25 % na snížení kvality. V ostatních případech mají 10 % na zvýšení kvality a 25 % na snížení.

**Generování výsledných parametrů**:

„item“ - dát generování šanci vygenerovat předmět (není tedy obsažen v každé odměně).

„cubeCoins“; „experience“ – dle statických vzorců a pravidel pro generování (balance – GenericDB/Balance).

„gold“ – pouze v případě, že boolean parametr „boss“ je true – tedy se jedná o odměnu za Boss fight.

„cubix“ – nativní generování bych nedělal, v budoucnu bude pouze externí generování, parametr „cubix“ bude v objektu Reward tedy statický.

**Snížení výsledných parametrů** – pouze v případě, že parametr „item“ není null, tedy nebyl vyygenerován, či funkce pro jeho generování selhala.