PREPA SIMULATOR

Cheat Sheet : Écriture de scripts

Motivation :

Jusqu'à la v0.2.0, le jeu était relativement linéaire : le joueur pouvait certes se déplacer sur une carte et parler aux PNJ, mais ces PNJ avaient des dialogues uniques et disaient donc uniquement ce qu'on leur indique de parler (ie. le texte inclus dans la BDD). L'implémentation du Sac a certes permis un nouveau degré de liberté pour le joueur, mais un nouveau moyen plus poussé d'interagir avec les PNJ était nécessaire.
L'implémentation du Scripting ~~rend à présent le jeu logarithmique~~ autorise une plus grande liberté dans les interactions possibles.

Structure du ScriptManager :
- un accumulateur booléen
- un accumulateur numérique
- un accumulateur liste, utilisé pour les infobox

Comment écrire son propre script ?
Il faut d'abord créer une nouvelle entrée dans la BDD, section « scripts ».
Ensuite, définir une nouvelle fonction du même nom dans le fichier `scripts.py`.
La fonction doit retourner une liste composée des instructions sous forme de chaîne de caractères. Préférer les doubles guillemets triples. Et surtout : NE JAMAIS UTILISER LES SIMPLES GUILLEMETS TRIPLES. ~~C'est moche.~~

Liste des fonctions et exemples :
※ *Fonctions générales*
- `runscript(script)`
Exécute le script dont le nom est donné en argument sous forme de chaîne de caractères.

※ *Fonctions booléennes*
- `compare_obj_qty(obj_id, operator, qty)`
Compare la quantité actuelle de l'objet ayant pour ID `obj_id` à la quantité `qty`.
Le résultat de la comparaison, stocké dans l'accumulateur booléen, dépend de l'argument `operator` :
`sup` >=
`inf` <=
`eq` =

- `iftrue(command)`, `iffalse(command)`
Exécute la commande `command` si l'accumulateur est `True` (resp. `False`).
`command` doit être une chaîne de caractères. Préférer les doubles guillemets simples.

Ex :
```
compare_obj_qty(0, sup, 50)
      iftrue(…)
```
fait quelque chose si le joueur dispose de plus de 50 exemplaires de l'objet d'ID 0.

※ *Fonctions numériques*

- `ran(inf, sup)`

Génère un nombre entier aléatoire dans l'intervalle [`inf`, `sup`]. Le résultat est stocké dans l'accumulateur numérique.

- `compare(operator, qty)`

Comme `compare_obj_qty`, mais avec des nombres.

Ex :

```
ran(1, 6)
compare('eq', 1)
      iftrue(...)
```

simule le lancer d'un dé et fait quelque chose si le résultat est égal à 1.

- `put(value)`

Place une valeur dans l'accumulateur, indépendante de sa valeur précédente.

- `math(operator, value)`

Effectue l'opération souhaitée avec l'accumulateur, et y place le résultat. `operator` est un caractère, les opérateurs supportés pour le moment sont `+ - * /`
Par exemple, `math('*', 10)` avec une valeur stockée de 15 place dans l'accumulateur la valeur 150.
En cas de division par 0, une valeur arbitrairement grande (sans souci de signe) est mise par défaut.

※ *Fonctions sonores*

- `chg_music(track)`

Change la musique courante en celle spécifiée en argument.

- `sfx(fx)`

Joue l'effet sonore mis en argument.

※ *Fonctions des flags des PNJ*

Important : *Les PNJ disposent pour le moment de deux flags seulement (la numérotation commence bien sûr à 0).*

- `checkflag(npc, flag_id)`

Place la valeur du flag `flag_id` du PNJ spécifié en argument, dans l'accumulateur booléen.

- `setflag(npc, flag_id)`

Modifie la valeur du flag `flag_id` du PNJ spécifié en argument.

※ *Fonctions des objets*

- `get_object(obj_id, qty)`

Le joueur obtient l'objet d'ID `obj_id` en quantité souhaitée.

- `toss_object(obj_id, qty)`

Le joueur perd l'objet d'ID `obj_id` en quantité souhaitée. Perdre plus que la quantité actuelle de l'objet ne lève pas d'erreur spécifique, le joueur perd juste tous les objets restants. Utiliser ce script librement lors de dialogues avec Lentsch.

※ *Fonctions des boîtes de dialogue*

- `dialogue(talking, dialogue_id)`

Engage avec le PNJ `talking` son dialogue associé d'ID `dialogue_id`.
ATTENTION, `talking` est un objet de type `Npc` ! Si le dialogue est engagé avec le PNJ auquel le joueur parle, utiliser `self.current_npc` à la place.

- `loadtext(text)`

Charge dans l'accumulateur associé une ligne du texte de l'infobox. Préférer pour `text` les simples guillemets simples : `'blabla'`

- `infobox()`

Génère une infobox en utilisant tout le texte présent dans son accumulateur, puis détruit le contenu de ce dernier.

※ *Fonctions de mouvement*

- `movingscript(direction, pix, sprint = False)`

Engage un mouvement de direction `direction` et de longueur `pix`, exprimée en pixels. L'argument `sprint` détermine si le joueur est en train de sprinter.
Les directions supportées sont :

```
up
right
down
left
```

Le mouvement est interrompu lorsque le joueur heurte un mur ou utilise un warp. ~~Après si vous voulez qu'il continue indéfiniment à marcher, il suffit de demander ;)~~

À titre d'exemple final, voici un script engageant un dialogue avec le PNJ à gauche de la I104 qui change si le joueur possède ou non l'objet d'ID 1, puis faisant sortir le joueur de la salle :

```
compare_obj_qty(1, 'sup', 1)
      iffalse('dialogue(self.current_npc, 1)')
      iftrue('dialogue(self.current_npc, 2)')
movingscript('up', 60)
movingscript('right', 800)
```