# Nomenclature

|  |  |  |
| --- | --- | --- |
| **Nom** | **~= RPGMaker** | **Description** |
| CheckPoint | Interrupteur | Booléen utilisé pour les conditions d’éxecution de groupes d’action |
| Counters | Variables | Valeur chiffrée utilisée pour les calculs ou conditions. |
| MapObject | Evenement | Objet positionnés sur la map, effectuant des groupes d’actions selon les conditions d’execution |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Sauvegarde

Cette fonction va être la partie la plus compliquée à implémenter et optimiser ! Il faut réfléchir durement à la chose pour ne pas avoir une sauvegarde qui duplique les éléments du jeu à chaque sauvegarde !

Liste des éléments à sauvegarder :

* **Monstres (toutes stats, attaques, états,…) :**
  + MonsterCollection (monstre rencontrés et capturés)
  + Monstres du Player
* **Player :**
  + Liste des items
  + Gold
* **World**
  + Map en cours + position
  + Tous les Checkpoints !

La sauvegarde se fera sous forme de Bases de données (BDD) ! Les fichiers seront dupliqués pour chaque sauvegarde.

On pourrait même définir une BDD System et une BDD sauvegarde qui communiqueraient les unes entre elles.

Dans un premier temps, on ne fait qu’une BDD et on séparera par la suite pour gagner en performances.

# Checkpoints

Pour simplifier la sauvegarde, pour ne pas avoir à sauver les MapObjects, on ne va pas implémenter les interrupteurs internes.

RPGMaker avait ce principe que chaque MapObject avait quatre « interrupteurs » (qu’on appellera Checkpoints) (A à D) pour spécifier ses différents états. Il s’agissait de 4 OnOff qui étaient surtout géré avec un ordre de priorité selon les listes d’actions.

On pourrait reprendre ce principe mais le sujet reste ouvert à la réflexion.

Vu que nos MapObjects vont normalement être exportés sous forme de fichier texte (fusionné dans les Maps),