

Death overhaul - guide d'installation

Version 2.0 – code original (PSDK) : Nuri Yuri

Code pour la gestion des zones : anime

Code moddé et event-making : Altarax, Amras Anárior

Documentation et installation :

Ce mod est lourd (de 24 à 28 plugins !) et possède donc un risque de conflit non négligeable si vous avez installé d'autres plug-ins, je vous conseille de faire un back-up de votre projet.)

Afin de limiter les risques de conflits, j'ai réussi à coder une détection du Nuzlocke natif de PSDK et de mes autres mods afin d'assurer une compatibilité automatique.

Compatibilité du mod

- **PSDK** versions **24.47** à **24.51** (v1 du Nuzlocke vanilla)
- **PSDK** versions **24.53** et futures versions 24 (v2 du Nuzlocke vanilla)
- Le mod « **Loyalty Overhaul** » de Sacred Phoenix
- Le mod « **Money of Keltios** » de Sacred Phoenix
- Le mod « **Overhaul de la capture** » de Sacred Phoenix (dans sa dernière version de mars 2020 – mod fourni en plugin optionnel dans le dossier « 03000 Balls »)

Features incluses dans le mod :

- **Mode Nuzlocke entièrement personnalisable**
 - Clause de la capture unique par zone indépendante de celles de mort si KO (Le maker peut au choix les jumeler ou les différencier.)
 - Clause chromatique
 - Clause doublon
 - Clause légendaire
 - Clause Black-Out
 - Clause objets interdits en combat (plugin optionnel, cf. plus bas)
- **Marquage des morts avec un statut « mort » distinct du KO.**

Permet de conserver ses Pokémon décédés dans une boîte cimetière.
- **Système de mort réaliste à mi-chemin un jeu Pokémon traditionnel et un Nuzlocke**

Offrant une difficulté intermédiaire entre ces deux modes, un Pokémon meurt s'il perd en un coup le double de ses PV max ([explications dans ce PDF](#)). Ce mode permet aussi à un Pokémon d'être « gravement KO » s'il a frôlé la mort, ce qui nécessitera des soins plus poussés.

• **Système de réincarnation et résurrection propre à Sacred Phoenix** ([explications dans ce PDF](#)), activable de trois manières différentes :

– **Mode 1 (par défaut)** : Résurrection possible durant un temps limité avec les Cendres sacrées (à l'exception des Pokémon spectres), puis chance de réincarnation en un Pokémon spectre à l'issue du cooldown. (Si échec de la réincarnation, ça devient une permadeath.)

– **Mode 2** : Résurrection possible sans limite de temps avec les Cendres sacrées (sauf spectres), mais pas de réincarnation.

– **Mode 3** : Vraie permadeath : pas de méthode de résurrection via les Cendres Sacrées, ni de réincarnation.

Dans le cas n°1, les Légendaires peuvent se réincarner en œuf et réapparaître dans l'équipe du joueur ou à un endroit déterminé sur l'une de vos maps.

Dans les cas n°1 et n°2, les Pokémon spectres subissent un bannissement en cas de « mort », ce qui équivaut à une permadeath.

Dans tous les cas, la résurrection peut toujours être forcée via une commande script insérable en événement-making. (Utile si vous souhaitez permettre la résurrection des Pokémon spectres par exemple.)

• **Poison à l'ancienne**

Si activé, un Pokémon qui tombe à 0 PV in-map deviendra KO comme dans les trois premières générations, au lieu d'automatiquement guérir avec 1 PV. Le KO sera bien entendu une mort si le mode Nuzlocke est activé.

Plugins optionnels (peuvent être utilisés en standalone) :

• **Mode de difficulté**

Offre 3 modes de difficulté permettant de modifier le niveau ($\times 0,9$; $\times 1$ ou $\times 1,1$) des dresseurs adverses. (Possibilité de créer d'autres paliers de difficulté ou de modifier les coefficients en allant dans le script.)

• **Objets interdits en combat**

Permet au joueur de se fixer une clause d'interdiction des objets en combat.

Permet aussi au maker de temporairement bloquer l'accès au sac pour un combat scénarisé. Plusieurs phrases types notifiant l'interdiction sont incluses de série.

• **Loyalty Overhaul v1**

Inflige une pénalité d'amitié lorsque le Pokémon tombe KO. La pénalité est proportionnelle aux PV perdus (toujours minimale en mode traditionnel, progressive en mode réaliste et toujours maximale en mode Nuzlocke).

Permet aussi à la Luxe Ball, à la Copain Ball et au Grelot Coque de fonctionner correctement.

Installation :

- Si vous avez une ancienne version de Death Overhaul, supprimez les anciens scripts.
- Décompresser l'archive.
- Mettre les dossiers "Data", "graphics" et "scripts" à la racine de votre Projet PSDK. Remplacer les fichiers, si demandé.
- Si vous souhaitez la feature de **difficulté** et/ou **d'objets interdits en combat**, installez respectivement dans votre dossier /scripts les fichiers :
16031 Difficulty level.rb
16101 Objets interdits.rb
Ils sont situés dans le dossier « -- plugins optionnels -- » du mod.
- Si vous souhaitez activer l'ébauche de **Loyalty overhaul**, installez dans votre dossier /scripts les fichiers :
12183 Loyalty overhaul - Bonheur max.rb
90002 Loyalty overhaul - pertes KO.rb
Ils sont situés dans le dossier « -- plugins optionnels -- » du mod.
- Renseignez les variables, constantes, arrays et interrupteurs nécessaires à ce mod dans le script maître : « 90000 Death overhaul - Gestion cycle vie-mort.rb ». (Explication plus bas dans la partie « Paramétrage du mod dans le Script maître ».) Tout est centralisé ici depuis la version 2.0 du mod.
=> Si vous n'utilisez pas certaines features, il ne sera pas nécessaire de tout renseigner.
- En vous aidant des screenshots et fichiers contenus dans "-- Events Communs à mettre sur RMXF --", créer les 3 événements communs demandés.
 - Un compte à rebours qui gère les réincarnations 3 jours après la mort. (Vous pouvez faire une version simplifiée qui ne gère que les jours écoulés.) (Inutile si vous n'utilisez pas la feature de réincarnation avec cooldown)
 - L'appel des textes des morts. (Inutile si vous n'utilisez pas la feature de réincarnation avec cooldown)
 - Éditer les deux lignes conditionnelles de l'évènement commun n°3 conformément au screenshot « Event commun 3 de retour au centre ». À noter que vous pouvez tout à fait créer votre propre évènement de retour au centre Pokémon ainsi qu'un Event commun de Game Over personnalisé. Pour vous inspirer, j'ai fourni mon script « 15164 Black out overhaul.rb » avec mon CommonEvents.rxdta. (Mon Black-out est géré en événement commun 54, 55 et 56.)
=> N'hésitez pas à vous aider des screenshots fournis et du fichier CommonEvents.rxdta (Vous pourrez ainsi copier-coller facilement et individuellement mes événements communs d'un projet "test" vers votre projet.)

- Dans votre projet, créer un ou plusieurs événements pour pouvoir modifier les variables et interrupteurs liés au mod et tester les changements. :)

- Profitez !

Paramétrage du mod dans le Script maître :

Tout se gère dans l'unique script : 90000 Death overhaul - Gestion cycle vie-mort.rb

Le code est commenté pour vous guider.

Interrupteurs RMXP :

Si vous n'utilisez ou ne proposez pas la feature correspondante au joueur, faire pointer le n° du switch sur un interrupteur qui restera toujours sur "false" (désactivé).

EnemyDeath

Mortalité des Pokémon adverses. Valeur à "true" si on peut tuer l'ennemi, "false" si on ne peut pas. Activer cette feature n'a (pour le moment) aucune conséquence sur le gameplay, mais le joueur saura avec quelle marge il aura mis KO (ou tué) le Pokémon adverse grâce au remplissage en négatif de sa barre de PV.

AreaClause

Permet d'activer de manière indépendante la Règle d'or n°1 du Nuzlocke : un seul Pokémon capturable par zone. Cette différenciation peut être utile si vous voulez permettre au joueur d'activer un mode « semi-Nuzlocke », ou avec des règles de capture personnalisées (ChainLocke, EggLocke, MonoLocke...)

Si vous souhaitez la jumeler avec la clause de mort des Pokémon (probable si vous voulez éviter de complexifier les choix offerts au joueur), faites comme suit dans votre événement :

Liste des commandes d'événement :

```
<> Message : Salut, tu veux activer le Nuzlocke ?
<> Choix : Oui, Non
: Si [Oui]
  <> Interrupteur [0164: Area Clause] = Activé
  <> Variable [0124: Mode Nuzlocke] = 2
  <> Variable [0125: Backup Nuzlocke] = 2
  <>
: Si [Non]
  <>
: Fin Choix
```

DupesClause

Permet d'activer la clause doublon. (Si le premier Pokémon rencontré dans une zone est déjà dans le Pokédex, la zone n'est pas bloquée malgré l'AreaClause.)

Son activation est sans effet en l'absence de l'AreaClause.

Si vous souhaitez ne pas proposer ce choix indépendamment du Nuzlocke, je vous conseille de jumeler cet interrupteur avec l'AreaClause. (Plus de 50 % des Nuzlockeurs choisissent d'appliquer cette clause.)

ShinyClause

Permet d'activer la clause chromatique. (Si un shiny est rencontré, il peut être capturé sans restriction malgré l'AreaClause.)

Son activation est sans effet en l'absence de l'AreaClause.

Si vous souhaitez ne pas proposer ce choix indépendamment du Nuzlocke, je vous conseille de jumeler cet interrupteur avec l'AreaClause. (Plus de 50 % des Nuzlockeurs choisissent d'appliquer cette clause.)

LegendaryClause

Permet d'autoriser la capture des Pokémon légendaires en Nuzlocke, même s'ils ne sont pas le Pokémon de zone au moment de la rencontre. Dépendant de l'array **LEG**.

Son activation est sans effet en l'absence de l'AreaClause.

Si votre scénario nécessite (ou recommande fortement) la capture d'un Légendaire, je vous conseille de jumeler cet interrupteur avec l'AreaClause.

BlackOutClause

À activer pour que la mort de tous les Pokémon de l'équipe du joueur aboutisse directement sur un Game Over, même s'il reste des survivants au PC.

L'activer systématiquement (choix fait par *Pokémon Uranium*) rend inutile le fait de coder la scène de retour au centre Pokémon en cas d'équipe morte.

OldSchoolPoison

Permet d'activer le poison à l'ancienne, c'est-à-dire d'avoir le Pokémon KO dès qu'il lui reste 0 PV in-map (ou mort si le Nuzlocke est activé).

Attention : N'ayant pas trouvé de solution universelle (glitch sur la commande scriptée de téléportation), ce script n'inclut pas le retour automatique au centre Pokémon si toute l'équipe est KO ou morte. Personnellement, j'ai rajouté ces lignes à la fin de la méthode dans le script : [12012 Death overhaul - Old school poison.rb](#)

```
74   if $pokemon_party.pokemon_revivable == 0
75     $game_temp.common_event_id = 54
76   end
```

L'événement commun 54 me permet de tester à minima s'il y a Game Over. (Vous pouvez aussi créer un événement commun custom qui fait retourner le joueur au point de repli, puis appelle l'événement commun n°3. Faites alors correspondre le *common_event_id = 54* vers votre événement custom.)

Variables RMXP :

Si vous n'utilisez ou ne proposez pas la feature correspondante au joueur, faire pointer le n° de la variable sur une qui restera toujours à "0" (zéro).

TimeElapsed

Mettre ici la variable utilisée pour mesurer le nombre de jours écoulés. Utilisée par la feature de réincarnation. À noter que si vous mettez **REINCARNATION = true** (cf. constantes ci-bas), mais que **TimeElapsed** pointe sur une variable qui reste figée ; la résurrection par les Cendres Sacrées sera possible sans limite de temps, mais les mécanismes de réincarnation n'auront pas lieu.

La vitesse d'incrémentation de cette variable se fait à votre discrétion. Vous pouvez créer un mécanisme d'horloge interne en suivant ce tuto (n'hésitez pas à simplifier le mécanisme si besoin, seule la variable « jours écoulés » est nécessaire).

<https://www.sacredphoenix.fr/forum/threads/syst%C3%A8me-de-baies-%C3%A0-la-2g-avec-horloge-interne-et-saisons.24/>

Difficulty

Utilisée dans le script "**16031 Difficulty level.rb**". -1 en mode facile, 0 en mode normal, 1 en mode difficile, ce qui correspond respectivement à 90%, 100% et 110% du niveau normal pour les Pokémon des dresseurs. Possibilité de rajouter des paliers supplémentaires ou de changer les coefficients en éditant le script.

Si le script **16031 Difficulty level.rb** est utilisé de manière standalone sans Death Overhaul, le n° de variable est à renseigner directement en ligne 10 dans le script en question.

Mortality

Mortalité de vos Pokémon. Valeur à 0 si désactivé, 1 si réaliste (mort si perte du double des PV max), 2 si mort immédiate dès le KO.

Mettre sur 2 permet d'appliquer la Règle d'or n°2 du Nuzlocke : à combiner avec l'activation du switch AreaClause pour avoir un Nuzlocke complet, mais je permets d'avoir ces deux règles indépendantes si jamais un maker veut instaurer la possibilité d'avoir des Nuzlockes aux règles spéciales (ChainLocke, EggLocke, MonoLocke...)

NoItemOnBattle

Permet de restreindre l'utilisation des objets en combat. 0 = Pas de restriction (vanilla), 1 et + : divers niveaux de restrictions avec un message différents (cf. fin du PDF).

Si le script **16101 Objets interdits.rb** est utilisé de manière standalone sans Death Overhaul, le n° de variable est à renseigner directement en ligne 13 dans le script en question.

Constantes et Arrays :

LEG

Array

Cette array est utilisé pour la **LegendaryClause** et la réincarnation.

Etablir les ID de vos Pokémon légendaires séparés par des virgules dans l'Array en ligne 10

Exemple : LEG = [245, 246, 248]

L'array est prérempli avec les légendaires et fabuleux jusqu'à la 7G + Meltan et Melmetal. À personnaliser si vous utilisez un Fakedex.

La particularité des Pokémon légendaires est qu'ils renaissent sous la forme d'un œuf soit directement dans votre équipe (si le test de réincarnation est réussi), soit quelque part dans la nature (toujours sous forme d'un œuf – ceci est paramétré à travers la variable EGG_POS).

REINCARNATION = true

Booléen

Mettre "true" + combiner avec une variable **TimeElapsed** fonctionnelle pour activer un stade temporaire de mort réversible où la résurrection via les Cendres Sacrées est possible la feature de réincarnation lorsque le **SOUL_COOLDOWN** expirera.

Mettre "true" + combiner avec une variable **TimeElapsed** figée pour permettre la résurrection via les Cendres Sacrées sans limite de temps tout en désactivant la réincarnation.

Mettre "false" pour avoir une vraie permadeath. (La réincarnation est forcément désactivée dans ce cas-là.)

Choisir l'une des deux dernières options permet de se dispenser de deux événements communs ainsi que du paramétrage de toutes les constantes qui suivent.

*-- Les constantes qui suivent ne sont pas à paramétrer si **REINCARNATION** = false ou **TimeElapsed** figée --*

SOUL_COOLDOWN

Nombre entier

Jours entre mort et mort définitif (paramétrable)

Réglée à 3 jours par défaut.

Combiné à l'événement commun d'écoulement du temps, cela permet de fixer un compte à rebours avant la mort définitive. Si l'écart entre le **TimeElapsed** et le timestamp de décès qui flag le Pokémon mort est supérieur ou égal au **SOUL_COOLDOWN**, la mort devient définitive.

COMMON_EVENT_DEATH_TEXTS

Nombre entier

Numéro de l'événement commun utilisé pour l'affichage des textes de réincarnation.

Constante EGG_POS

Array 2D

Double array qui indique les maps et les événements de repos des œufs des légendaires qui ont trépassés (mais n'ont pas voulu se réincarner tout de suite auprès du joueur).

Confer la section « *Événements de la renaissance sous forme d'œuf des Légendaires* » pour l'explication en détail du fonctionnement.

Constante GHOST_ID

Array

Éléments d'array sous la forme {276 => 292, 278 => 294}, chaque paire de valeur étant séparé par une virgule.

Cet array détermine en quoi un Pokémon mort (premier chiffre) est susceptible de se réincarner (second chiffre = ID du Pokémon de réincarnation). (Prioritaire sur la réincarnation par type.)

Constante GHOST_TYPE

Array

Éléments d'array sous la forme {5=>708, 3=>592}, chaque paire de valeur étant séparé par une virgule.

Le premier chiffre est le type du Pokémon et le second l'ID du Pokémon spectre en lequel il est susceptible de se réincarner.

Constante GUARANTEE_REBIRTH

Array

Array des ID des Pokémons qui auront une probabilité de 100% de réincarnation après l'expiration du **SOUL_COOLDOWN** (à la demande de Zenos).

Se marie très bien avec la gestion des formes. (Cas d'un Pokémon précis qui doit se réincarner sous sa forme spectrale en cas de mort.)

Paramètres au cœur de la méthode :

def resurrect_id(pokemon)

Dans le cas où ça ne trouve ni correspondance par Pokémon précis, ni par type, ça tire au hasard un des 3 Pokémons sceptres listés dans ces lignes

```
r = rand(3)
return 200 if r==0
return 353 if r==1
return 355
```

La liste peut être allongée ou changée en rajoutant des lignes return et en faisant correspond le numéro x du rand(x) par le nombre de return (id) if r==(rand).

def resurrect?(pokemon)

Cette partie détermine la probabilité de réincarnation d'un Pokémon mort 3 jours après sa mort. Dans Sacred Phenix, elle dépend de l'amitié (après prise en compte du gros malus de -62/255 inhérent à la mort subie.)

- Mettre « return true » (déneutraliser la ligne en commentaire) pour avoir 100% de probabilité de réincarnation.

- Pour qu'un Pokémon précis ait 100% de probabilité de réincarnation, lister son ID dans l'array de la constante **GUARANTEE_REBIRTH**.

- À noter que tout Pokémon non Légendaire de type Spectre ne peut se réincarner : un « mort » qui remeurt à nouveau est banni à jamais du monde des vivants et passe directement en état « mort définitive ». (D'ailleurs, le jeu dit bien que le Pokémon Spectre qui tombe au combat est « banni » au lieu de « succombe »)

Gestion des formes

Pour gérer les formes (et faute de savoir comment scripter un array gérant cela), je me suis incrusté directement dans la méthode à cet endroit :

```
96     pokemon.id = resurrect_id(pokemon)
97     # ----- Gestion des formes spéciales (je m'incruste ici) -----
98     if pokemon.id == 105 or pokemon.id == 222
99         pokemon.form = 1 # Forme spectrale (Ossatueur d'Alola / Corayon de Galar)
100     end
101     $names_s.push(pokemon.given_name)
```

La technique, c'est que si le Pokémon réincarné a tel ID (ici 105 : Ossatueur ; ou 222 : Corayon), je force sa forme n°1 (Ossatueur d'Alola ou Corayon de Galar) qui sont tous deux de type Spectre.

(Bonus) Paramètres de Loyalty Overhaul :

Si vous ne voulez pas du petit malus d'amitié avec certains objets "maléfiques", il suffit de supprimer les 4 lignes de code contenant le mot " i.loyalty" dans le script : **16150 Patch Scene Battle BEGEND Turn.rb**

Ce script (obligatoire) patche un bug avec l'Orbe Vie lié à l'apparition des PV en négatif. Il y a quelques légers rééquilibrages sur les dégâts ou la régen de certains objets tenus + un patch pour éviter l'affichage du texte de régen lorsque le Pokémon est déjà full HP.

Commandes de scripts ajoutées par le mod :

Commandes pour soigner les Pokémon

Actuellement, 3 niveaux de « soins » sont codés dans le mod. Bien entendu, les morts et morts définitifs sont exclus du soin.

\$pokemon_party.full_heal_party

Soigne toute l'équipe entièrement, y compris les « KO au bord de la mort ».

\$pokemon_party.heal_party

(Commande native de PSDK) Soigne toute l'équipe entièrement, sauf les « KO au bord de la mort » qui passent à un stade de KO normal. Appeler deux fois cette commande revient à utiliser la première commande.

\$pokemon_party.partial_heal_party

Regénère tous les Pokémon non KO à hauteur de 50 % de leurs PV et de 33 % de leurs PP. Les problèmes de statuts sont soignés, sauf les KO et KO au bord de la mort qui perdurent.

À terme, je prévois un mécanisme de soins « à la RPG old-school » où le soin s'incrémente d'un pourcentage pour chaque « heure » passée à se reposer. Ce mécanisme est adapté à mon univers médiéval-fantastique.

La commande « **\$pokemon_party.heal_party** » restera bien entendu toujours disponible pour les machines « modernes » qui offrent un soin instantané.

Test pour savoir si un Pokémon est plus ou moins mort :

\$pokemon_party.actors[0].mort?

\$pokemon_party.actors[0].mort_d?

\$pokemon_party.actors[0].damaged?

...pour savoir si le premier Pokémon est respectivement :

- mort
- mort définitivement
- ou K.O. au bord de la mort

La commande native de PSDK “.dead?” englobe les statuts KO, mort, et mort déf, aka si le Pokémon n’est plus apte au combat.

Test pour savoir si l’équipe du joueur est plus ou moins morte :

\$pokemon_party.pokemon_revivable

Retourne -1 si :

- L’équipe est vide
- L’équipe ne comporte que des œufs

Retourne 0 si tous les Pokés autres que œufs sont morts (ou morts définitivement).

Sinon, retourne le nombre de Pokémon aptes à être soignés via les moyens conventionnels.

\$pokemon_party.allmort?

Booléen : teste si toute l’équipe du joueur est morte.

Attention : ici, un œuf est considéré comme un Pokémon vivant. (Ce qui au fond n’est pas faux.)

\$pokemon_party.fainted_pokemon

Total du nombre de Pokémon KO (KO simple) dans l’équipe du joueur

\$pokemon_party.damaged_pokemon

Total du nombre de Pokémon KO au bord de la mort dans l’équipe du joueur

\$pokemon_party.out_of_battle

Total du nombre de Pokémon KO ou KO au bord de la mort dans l’équipe du joueur

Pour ressusciter le premier Pokémon :

Dans une commande script, entrer ces trois commandes :

\$pokemon_party.actors[0].status = 0

\$pokemon_party.actors[0].hp = \$pokemon_party.actors[0].max_hp

\$pokemon_party.actors[0].death_c = nil

Le second guide d’installation axé sur l’évent-making vous donnera des exemples plus concrets.