

Les options dans Sacred Phoenix (version PSDK 2023)

(Version PSDK 24.87 ultra moddée)

Localisation du code des options

Depuis la version 24.49 de PSDK, les options ne sont plus codées en dur dans les scripts Ruby ni sur la partie « scripts » de RPG Maker mais dans le fichier `\Data\project_indentity.yml`. C'est aussi ici qu'on peut rajouter de nouvelles polices d'écriture ou changer la résolution du jeu (plusieurs constantes y sont référencées.)



La partie « options » du `project_indentity.yml` a été court-circuité avec succès en faisant une pirouette de class dans `00001 Custom game configuration.rb`. De ce fait, j'ai pu supprimer pas mal de lignes de ce fichier moins malléable qu'un script ruby.

Donc si on crée une option supplémentaire, la rajouter dans `00001 Custom game configuration.rb`. (Et non `project_indentity.yml` qui n'en gère plus aucune)

Si malgré cela, des changements doivent être fait au `project_indentity.yml`, il est important de le convertir en ".rxdata" en lançant le jeu depuis le CMD puis en rentrant « game » en ligne de commande, car seul le fichier .rxdata fait foi pour le jeu.

Liste des options dans PSDK vanilla (état actuel)

Pour rappel, les options actuelles de PSDK vanilla sont :

- **Vitesse du texte** (opérationnel)
- **Fenêtre** (opérationnel, non désiré)
- **Volume** (opérationnel)
- **Animation de Combat** (opérationnel)
- **Style de combat** (opérationnel)
- **Langue** (opérationnel)

Toutes étaient liés à la sauvegarde. Désormais, la langue et le volume sont liés à un `setting.ini`.

- **Style de combat** restera lié à la partie en cours (car ça fait partie d'un réglage de difficulté)
- **Vitesse du texte** et **Animation de Combat** seront associé au `setting.ini`.

Ajoutés :



- **Résolution** : lié au `setting.ini` et fonctionnel.



- **Divers réglages de difficulté** (Nuzlocke entre-autre) : fonctionnel, avec réajustement automatique des réglages qui dépendent d'un autre.

Confer : https://www.sacredphoenix.fr/res/Telechargements/PDF/GDD/Difficulty%20settings_FR.pdf

Objectifs souhaités pour Sacred Phoenix

Résumé rapide :



- **(Urgent)** : Retrouver le fonctionnement de la langue tel qu'il était dans Sacred Phoenix en version 24.48 (aka lié à tout le jeu et dépendant d'un fichier de configuration)

  (**Urgent**) : Réparer encore ce bug de non actualisation du menu du jeu à la sortie des options en cas de changement de langue.

  (**Urgent**) : Virer le choix des fenêtres de dialogue. (Sacred Phoenix ne proposera qu'un seul style : celui du parchemin.)

  (**Conseillé**) : Débugger le réglage de la BGM et le désolidariser du réglage des canaux ME/SE. (Donc 2 options de réglage du volume au lieu d'une seule.)

  (**Conseillé**) : Réparer le choix du style du combat « choix / défini » qui de toutes manières ne fonctionne pas. (La permission de switch gratuit ou non sera dictée dans 95 % des cas par le scénario du jeu et non par le joueur.) (Finalement je laisserai l'option pour les 5 % restants.)

  (**Conseillé**) : Intégrer dans les options le choix de la difficulté qui dépendent de variables et interrupteurs RMX. À l'heure actuelle, il en existe trois : **Niveau des dresseurs**, **Mort de vos Pokémon** et **Mort des Pokémon adverses**. Lorsque les clauses Nuzlocke seront implémentées, d'autres options se rajouteront.

  (**Bonus**) : Coder la gestion du style de combat (choix / défini) et permettre de toggle ce paramètre à travers un interrupteur RMX.

  (**Bonus**) : Permettre l'accès au menu « Options » depuis le menu principal (ce qui suit l'écran-titre). Corolaire : en cas d'absence de sauvegarde, le menu principal proposera « Nouvelle partie » et « Options ». (À terme, c'est de me rapprocher du fonctionnement de *Pokémon Uranium*.)

  (**Bonus**) : Jouer la musique « Intro - Main Menu » dans ce menu principal. (Comme pour Pokémon or et argent.)

  (**Bonus**) : Créer trois (ou quatre) sous-menus d'options : Réglages, Difficulté, Langue (et Commandes). Quand le joueur choisira « Options » depuis le menu du jeu (ou depuis le menu principal si c'est implémenté), un menu de second niveau proposera ces 3 ou 4 choix. Je peux gérer le menu de langue en full événement making si la scène appelle un événement commun.

  (**Bonus**) : Coder un slider pour le réglage de la difficulté globale et celle des 4 volumes sonores.

  (**Bonus**) : Débugger la reconnaissance manette et optimiser le menu de mapping des touches (celui appelé par F1)

Ce qui est « **Urgent** » est indispensable à faire pour que je puisse release une version propre de Sacred Phoenix.

Ce qui est « **Conseillé** » fera un peu plus propre et sera la bienvenue si vous avez le temps de vous en occuper, mais je peux release sans ces objectifs atteints.

Ce qui est « **Bonus** » est un bonus. Ce sont des objectifs de moyen et long-terme.

 les variables RPG Maker sont réinitialisées lors qu'une partie est chargée ou rechargée. Toute modifications de celles-ci depuis l'écran titre ou le menu principale ne seront donc pas conservées.

De ce fait, si les options sont appelées depuis le menu principal ou l'écran-titre, seuls les changements suivants seront à proposer :

- Langue
- Affichage (résolution + volume audio)
- Commandes

Les options de difficultés ne sont à proposer que dans une partie en cours.

Cheminement pour atteindre ces objectifs

1) Permettre aux options de se baser en écriture / lecture sur 3 types de variables :

– Des variables internes à la sauvegarde (fonctionnement actuel de toutes les options)

Cela sera propre à chaque sauvegarde.

~~En nous privant de coder en dur dans le code les mécanismes de lecture / écriture de chaque option via la centralisation sur le **project_identity.yml**, le panachage de méthodes est devenu impossible. Seul ce fonctionnement est possible sur PSDK Vanilla. Court-circuité~~

– Des variables et interrupteurs RPG Maker XP

Cela sera propre à chaque sauvegarde.

Utilisées notamment pour le paramétrage de la difficulté. C'est ce qui me permettra de rajouter des options custom le plus facilement, l'évent-making m'étant bien plus accessible que coder une nouvelle option en dur dans des scripts.

~~À noter qu'une fois toutes ses clauses implémentées à 100 %, le Mode Nuzlocke ne sera plus ré-activable depuis les options (seulement via des événements en début de jeu). L'inverse sera néanmoins possible. (Désactivable à tout moment, mais plus ré-activable s'il a été désactivé.)~~
Inutile maintenant, le jeu enregistre avec succès la difficulté la plus basse utilisée.

En attendant, le joueur pourra changer librement tous ses paramètres de difficulté depuis le menu.

– Des variables liées à tout le jeu, notées en clair sur un setting.ini

Cela sera commun à toutes les parties. La **langue** en fait partie de même que les **commandes**.

Je me tâte aussi de mettre le **réglage du volume** ici pour des raisons évidentes. (Éviter la musique assourdissante à l'écran-titre si le joueur a baissé significativement le volume au sein de sa sauvegarde.)

À noter qu'un fichier **input.yml** est présent dans le dossier "Saves" et renferme le mapping des touches clavier. Si c'est plus sécurisant, lire/écrire les paramètres des options sur ce format « yml ».

2) Trois (ou quatre) menus séparés :

Je me tâte potentiellement de faire 3 ou 4 menus séparés :

- un pour les options classiques nommé « **Réglages** » (vitesse texte, son, animation combat...)
- un pour les paramètres de gameplay nommé « **Difficulté** ». Correspond aux modes de mortalité des Pokémon, à la, force des dresseurs et toutes les futures clauses de Nuzlocke.
- un pour la langue nommé « **Langue** ».
- Et le quatrième qui pourrait rejoindre cette famille s'appellerait « **Commandes** ». Il existe déjà : il suffit de presser [F1] en jeu pour y accéder, mais peu de joueurs le savent.

Quand le joueur sélectionnera « Options » depuis le menu du jeu, un sous-menu avec ces trois ou 4 embranchements apparaîtra.

Le fait de séparer en 3 (ou 4) menus est lié aux trois types de variables possibles que je souhaite pour des options.

Le réglage du volume fera probablement exception, car je souhaite le rendre commun au jeu en entier (histoire que le volume soit au bon niveau dès le démarrage du jeu).

3) Optimiser la gestion de la langue :

On m'a dit qu'un fichier ini comporte un risque de sécurité.

Dans le dossier « Saves », j'ai constaté la présence d'un « input.yml » qui est utilisé pour le mapping des touches. (Mapping commun à tout le jeu et non une sauvegarde spécifique... on notera d'ailleurs que le menu de mapping des touches est distinct du menu

Si c'est plus approprié : **passer au format YML pour le paramétrage de la langue.**

J'ai constaté aussi que 2 variables sont utilisées pour un changement de langue sur PSDK moddé (contre une seule en version vanilla).

À voir si une seule pourrait suffire.

Le choix de la langue Réalisé

Désormais enregistrée dans **setting.ini** et **\$Options.language** se synchronise systématiquement dessus. Si le ns **setting.ini** n'existe pas (ou que la ligne de langue n'existe pas), il est automatiquement créé avec la langue anglais par défaut.



Un objectif bonus serait que si la langue n'est pas encore définie, qu'elle soit automatiquement synchronisée sur celle du système d'exploitation si ce dernier est d'une langue existante autre que l'Anglais (en d'autres termes : français et espagnol actuellement). Si l'OS n'est pas d'une de ces langues, le **setting.ini** sera généré avec la langue anglaise comme réglage.

Les autres options

1) Volume

Options de volume à créer ou corriger : ( Réalisé)

- Volume master
- Volume musique (gère le volume des canaux BGM et BGS)
- Volume effets sonores (gère le volume des canaux ME et SE)
- Volume cri Pokémon (idée venue des remakes Pokémon DEPS)
- + Débarrer la gestion du volume musique qui ne fonctionne pas (non refresh des volumes).

D'après le code du Project Identity, le volume aurait dû se régler avec un slider (réglette), mais ça ne semble pas encore être programmé sur PSDK... À voir si ça peut se faire.

Quoi qu'il en soit, j'augmenterai le pas d'incrément de 1 à 5 %. (Il n'y a pas besoin d'une précision si fine.)

Ces paramètres doivent être codés dans le settings.ini (ou settings.yml si on part sur le format yml) et seront donc commun à tout le jeu. Cela permettra au réglage sonore d'être actif dès le démarrage du jeu et d'éviter la musique d'écran-titre assourdissante. (Toutes les réglettes sont à 50% par défaut, soit -12 dB).



Ne manque donc plus qu'à afficher sous forme de réglette plutôt qu'un chiffre (le chiffre peut rester visible sur l'écran, c'est même conseillé).

2) Style de combat



Réalisé

Tous mes idéaux ont été atteints.

Par défaut, ce réglage sera ignoré dans 95% des combats. Seul les combats « amicaux d'entraînement » (défini par une variable réglée manuellement par moi avant le combat) prendront en compte ce réglage.

Cette variable possède aussi un réglage « Choix forcé » pour les matchs imposant obligatoirement ce choix, même si cette option est sur « Off » (cas des champions d'arène.)

Le mode assisté permet d'avoir le comportement vanilla d'un jeu Pokémon (= autorisé sur tous les combats solos non spéciaux si l'option est sur « On »).

Du fait que ce réglage a un impact négligeable sur la difficulté, il n'a aucune influence sur le calcul de la difficulté globale.

3) Mode de difficulté

Je renvoie une fois de plus à ce document

https://www.sacredphoenix.fr/res/Telechargements/PDF/GDD/Difficulty%20settings_FR.pdf

Tout a été réalisé, ou presque.



Il me manque juste une chose : que le réglage de difficulté est un impact sur l'UI. Ceci est décrit à la fin du document.

Je souhaite notamment :

- Que sur le menu principal, une icône spéciale apparaisse sur le coin supérieur droit de la sauvegarde si la partie enregistrée a un Nuzlocke valide en cours ou est actuellement en mode assisté. (Un Nuzlocke est considéré comme valide si la difficulté la plus basse est égale au moins au preset Nuzlocke)
- Que les noms de nos Pokémon et de nos ennemis soient affichés dans une couleur légèrement différente sur l'écran de combat (inchangé en difficulté normale, teinte rosâtre si réglage qui y correspond est plus facile, teinte dorée si le réglage qui y correspond est plus difficile.)



Le menu de touche

Il s'agit de ma bête noire, car

- Il est peu ergonomique
- 5 colonnes d'association (dont 4 pour le clavier), c'est trop.
- La 5^{ème} colonne (celle de touche manette) est buguée et ne peut être associée (facteur aggravant : ça marche avec PSDK 26 qui est basé sur LiteRGSS 2, mais pas avec LiteRGSS 1 de PSDK 24). J'ai dû coder un anti-bug (message qui affiche tout le temps « manette non détectée » dès que le joueur tente de modifier cette colonne), sinon ça crashe.

- L'appeler à un mauvais moment peut causer un crash.
- Mais il est impossible de l'appeler tant qu'on n'a pas accès au menu de jeu (donc pas avant d'être libre de nos mouvements dans la Colline du Phénix.) Cela inclut l'écran titre !

TOUCHES		
Tch. PSDK	Description	Association
A	Valider, Parler, Agir	[C] [F4] [C] [A]
B	Annuler, Courir, Quitter	[X] [F4] [3] [F4] [B]
X	Ouvrir le menu	[V] [0] [V] [V] [X]
Y	Raccourcis	[W] [alt] [W] [W] [Y]
Haut	Déplacement haut	[↑] [Z] [8] [↑] []
Bas	Déplacement bas	[↓] [S] [2] [↓] []
Droite	Déplacement droite	[→] [D] [6] [→] []
Gauche	Déplacement gauche	[←] [Q] [4] [←] []

[C] : Chx. touche
[→] : Chx. touche
[↓] : suivant
[X] : quitter



Mes objectifs :

- Pouvoir appeler ce menu dès le menu principal, avant même de lancer une nouvelle partie., de même que depuis le menu option du menu de jeu.
 - Réduire à 2 colonnes les associations de touche clavier. (Corolaire : il faudra modifier le **input.yml** qui est généré par défaut lorsqu'il n'existe pas, car il génèrera 4 colonnes.)
 - Rajouter une 9^{ème} touche paramétrable qui correspond actuellement à la touche [Ctrl gauche], utilisé pour toggle le Follow Me. (Ce toggle n'est pas juste esthétique, ça des implications de gameplay). On peut l'appeler « L »
 - Rajouter une 10^{ème} touche paramétrable pour le Speed Up (touche [F5] par défaut). On peut l'appeler « R ».
 - Regarder l'ergonomie (j'ai constaté qu'il est difficile d'activer le re-paramétrage d'une touche : ça répond de manière erratique). Cela est dû qu'il faut faire minimum 3 actions pour remapper une touche :
 - Avec les flèches haut et bas, se placer à la bonne ligne.
 - Appuyer sur la touche « A » ([C] ou [Entrée] par défaut)
 - Puis sélectionner la colonne avec les touches droite et gauche
 - Puis encore appuyer sur la touche « A » ([C] ou [Entrée] par défaut)
 - La touche clignote : on peut en choisir une autre (ouf !)
- Supprimer l'étape du choix de colonne en l'automatisant serait une très bonne idée.

- Si l'input envoyé correspond à une touche de manette, modifier la colonne manette (donc la dernière)
 - Si c'est une touche clavier et la touche choisie est déjà paramétrée sur cette ligne, supprimer la seconde entrée (si elle existe) qui n'est pas de cette touche. (S'il est impossible de mettre une entrée vide, donner 2 fois la même valeur aux 2 colonnes de touche).
 - Si la touche clavier choisie est déjà attribuée à une autre action, changer cette touche mais attribuer l'ancienne valeur à celle d'où provient le doublon (on intervertit en gros).
- De plus, ce menu ne réagit pas à la souris. Par exemple, si je clique à la souris sur l'une des touches listées, je dois pouvoir la reparamétrer tout de suite et précisément cette touche si j'appuie sur une touche clavier.
 - Un clic droit devrait être équivalent à la touche « Annuler » ici. Et si la souris pointe sur une touche de la seconde colonne, effacer la dite touche.
 - Coder un check de double attribution de touche qui peut mener à un blocage (des joueurs ont dû supprimer leur **input.yml** à cause de ça. Idéalement en intervertissant l'ancienne touche avec celle qui est en doublon.