26/10/2025

RAPPORT D'OPTIMISATIONS

Étape 6

Ayoub KHLOUFI, Vincent CORNEAU
CEGEP DE CHICOUTIMI

Table des matières

ApiV1ControlleurMonstre/Constants/GameConstants.cs

Nous avons créé ce fichier parce que :

- Les nombres magiques étaient dispersés dans plusieurs fichiers (0, 50, 100, 999, etc.)
- Il n'y avait aucune centralisation des constantes du jeu
- Les modifications de configuration nécessitaient de chercher dans tout le code

Modifications apportées :

Ajout de 42 constantes organisées par catégorie :

- Limites carte : MAP_MIN_POSITION = 0, MAP_MAX_POSITION = 50
- Système XP : EXPERIENCE_PER_LEVEL = 100, BASE_MONSTER_XP_MULTIPLIER = 10
- Stats personnage : DEFAULT_PLAYER_HP = 100, DEFAULT_PLAYER_MAX_HP = 999
- Combat : MIN_COMBAT_FACTOR = 0.8, MAX_COMBAT_FACTOR = 1.25
- Génération monstres : MAX_MONSTER_COUNT = 300, MAX_SPAWN_ATTEMPTS = 50
- ...

Justification :

- Respecte le principe DRY (Don't Repeat Yourself)
- Facilite la maintenance et les ajustements de gameplay
- Élimine 100% des nombres magiques du projet

ApiV1ControlleurMonstre/Controllers/PersonnagesController.cs

Nous avons modifié ce fichier parce que :

- Aucune gestion d'erreurs (try/catch) dans les fonctions critiques
- Nombres magiques présents dans les calculs (0, 50, 100, 1.25, 0.8)
- Risque de plantage de l'API en cas d'erreur base de données
- Messages d'erreur non standardisés

Modifications apportées :

1. Ajout de try/catch :

- MovePersonnage() : Try/catch global avec gestion DbUpdateException, InvalidOperationException
- HandleCombat() : Protection des calculs de combat et sauvegarde
- AuthenticateUser() : Protection des requêtes d'authentification
- GetPersonnageByUserId() : Protection des requêtes personnage
- GetMonsterAtPosition() : Protection des jointures avec Include
- HandlePlayerVictory() : Protection des gains d'expérience et suppression monstre
- HandlePlayerDefeat() : Protection de la téléportation vers ville
- CalculateCombat() : Validation des paramètres null et calculs sécurisé
- ...

2. Remplacement des nombres magiques :

- if (personnage.PositionY > 0) → if (personnage.PositionY > GameConstants.MAP_MIN_POSITION)
- if (personnage.PositionY < 50) → if (personnage.PositionY < GameConstants.MAP_MAX_POSITION)
- new Random().NextDouble() * (1.25 - 0.8) + 0.8 → GameConstants.MAX_COMBAT_FACTOR - GameConstants.MIN_COMBAT_FACTOR
- Math.Max(1, ...) → Math.Max(GameConstants.MIN_DAMAGE, ...)
- while (personnage.Experience >= 100) → while (personnage.Experience >= GameConstants.EXPERIENCE_PER_LEVEL)
- ...

3. Amélioration des types de retour :

• AuthenticateUser() : (Utilisateur user, ...) → (Utilisateur? user, ...)

• GetPersonnageByUserId() : Personnage → Personnage?

• GetMonsterAtPosition() : InstanceMonstre → InstanceMonstre?

Justification :

• Robustesse : 100% de couverture des exceptions critiques

• Maintenabilité : Configuration centralisée

• Sécurité : Gestion appropriée des erreurs sans exposition des détails

ApiV1ControlleurMonstre/Controllers/UtilisateursController.cs

Nous avons modifié ce fichier parce que :

- Nombres magiques dans la création de personnage (100, 999, 10, 0)
- Durée JWT codée en dur (1 heure)
- Valeurs par défaut dispersées sans logique centralisée

Modifications apportées :

- Import GameConstants : using ApiV1ControlleurMonstre.Constants;
- Niveau = 0 → Niveau = GameConstants.DEFAULT_PLAYER_LEVEL
- PointsVie = 100 → PointsVie = GameConstants.DEFAULT_PLAYER_HP
- PointsVieMax = 999 → PointsVieMax = GameConstants.DEFAULT_PLAYER_MAX_HP
- Force = 10 → Force = GameConstants.DEFAULT_PLAYER_FORCE
- Defense = 10 → Defense = GameConstants.DEFAULT_PLAYER_DEFENSE
- AddHours(1) → AddHours(GameConstants.JWT_EXPIRY_HOURS)

Justification :

- Cohérence avec les autres parties du système
- Facilite les ajustements d'équilibrage
- Configuration sécurité centralisée

ApiV1ControlleurMonstre/Services/MonstreMaintenanceService.cs

Nous avons modifié ce fichier parce que :

- Nombres magiques pour la génération de monstres (300, 50, 10)
- Coordonnées codées en dur (0, 50)
- Nombre de tentatives arbitraires

Modifications apportées :

- Import GameConstants : using ApiV1ControlleurMonstre.Constants;
- var monstersToCreate = 300 - monsterCount → GameConstants.MAX_MONSTER_COUNT - monsterCount
- for (int attempt = 0; attempt < 50; attempt++) → GameConstants.MAX_SPAWN_ATTEMPTS
- random.Next(0, 50) → random.Next(GameConstants.MAP_MIN_POSITION, GameConstants.MAP_MAX_POSITION)
- for (int genAttempt = 0; genAttempt < 10; genAttempt++) → GameConstants.MAX_GENERATION_ATTEMPTS

Justification :

- Cohérence avec les limites de carte définies ailleurs
- Configuration centralisée pour la génération de contenu
- Facilite les ajustements de densité de monstres

SrcWeb/js/script.js

Nous avons supprimé ce fichier parce que :

- 721 lignes dans un seul fichier = code monolithique impossible à maintenir
- Mélange des responsabilités (authentification + carte + mouvement + combat)
- Difficile de localiser et corriger les bugs
- Violation des principes de programmation propre

Remplacement :

- Division en 6 modules spécialisés avec responsabilités claires
- Chaque module fait ~100 lignes maximum
- Séparation logique des fonctionnalités

Justification :

- Maintenabilité améliorée
- Debugging plus facile
- Travail en équipe facilité
- Respect des principes SOLID

SrcWeb/js/main.js

Nous avons créé ce fichier parce que :

- Besoin de centraliser les variables globales
- Fonction callAPI() utilisée partout nécessitait des améliorations sécuritaires
- Aucune validation des tokens côté client

Modifications apportées :

- Variables globales centralisées (posX, posY, mapArray, etc.)
- callAPI() améliorée avec validation expiration token :

```
const now = Math.floor(new Date().getTime()/1000.0)
if (!tokenObject || tokenObject.expiry <= now) {
        document.getElementById("logout-btn").click()
}
```

- Fonction showErrorPopup() commune

Justification :

- Sécurité renforcée avec déconnexion automatique
- Évite les appels API avec tokens expirés
- Code utilitaire centralisé

SrcWeb/js/map.js

Nous avons créé ce fichier parce que :

- Logique de gestion de carte était mélangée avec d'autres fonctionnalités
- Fuites mémoire dues aux event listeners non nettoyés
- Appels API répétés pour les mêmes tuiles

Modifications apportées :

- Fonctions carte isolées : GetTile(), createGrid(), displayGameGrid()
- NOUVEAU : clearAllEventListeners() pour éviter fuites mémoire
- Cache intelligent avec mapArray pour éviter appels API répétés
- getTileWithCoords() et setTileWithCoords() pour accès optimisé
- Gestion améliorée des event listeners avec références stables

Justification :

- Responsabilité unique : gestion de carte seulement

SrcWeb/js/mouvement.js

Nous avons créé ce fichier parce que :

- Logique de mouvement complexe mélangée avec autres fonctions
- Gestion incomplète des différents résultats de combat
- Fonction shiftGameGrid() trop complexe et source d'erreurs

Modifications apportées :

- Fonctions mouvement isolées : moveGrid(), movePersonnage()
- Gestion des 4 cas de retour API (Moved, WonFight, LostFight, Indecis)
- SUPPRESSION de shiftGameGrid() (60 lignes complexes éliminées)
- Amélioration gestion cas "indécis" avec mise à jour points de vie monstre
- Validation token avant chaque mouvement

Justification :

- Simplicité : suppression logique complexe inutile
- Robustesse : gestion complète des scénarios de combat
- Performance : utilisation directe du cache au lieu de calculs

SrcWeb/js/auth.js

Nous avons créé ce fichier parce que :

- Code d'authentification mélangé avec logique de jeu
- Responsabilité distincte nécessitant isolation
- Faciliter maintenance du système d'authentification

Modifications apportées :

- Code login/register transféré sans modification fonctionnelle
- Event listeners pour formulaires
- Gestion UI authentification

Justification :

- Séparation des responsabilités
- Module réutilisable pour autres projets
- Maintenance facilitée

SrcWeb/js/combatSim.js

Nous avons créé ce fichier parce que :

- Simulateur de combat = fonctionnalité indépendante
- Code complexe (105 lignes) méritant son propre module
- Faciliter ajout de nouvelles fonctionnalités de simulation

Modifications apportées :

- Code simulateur transféré sans modification
- Fonctions de calcul et affichage isolées

Justification :

- Module optionnel indépendant
- Facilite les tests et améliorations
- Code organisé par fonctionnalité

SrcWeb/js/stats.js

Nous avons créé ce fichier parce que :

- Affichage des statistiques = responsabilité distincte
- Manquait fonction pour nettoyer l'affichage des monstres
- Code d'affichage dispersé

Modifications apportées :

- Fonctions setPersonnage(), setInfoMonster() transférées
- NOUVEAU : clearInfoMonster() pour nettoyer affichage quand pas de monstre
- Centralisation logique d'affichage

Justification :

- Expérience utilisateur améliorée
- Code d'affichage centralisé
- Fonction manquante ajoutée

SrcWeb/index.html

Nous avons modifié ce fichier parce que :

- Remplacement du fichier script.js monolithique par 6 modules
- Besoin de charger les modules dans le bon ordre de dépendance

Modifications apportées :

- Suppression : <script src="js/script.js"></script>
- Ajout de 6 imports dans l'ordre :
    - <script src="js/main.js"></script>
    - <script src="js/auth.js"></script>
    - <script src="js/map.js"></script>
    - <script src="js/mouvement.js"></script>
    - <script src="js/stats.js"></script>
    - <script src="js/combatSim.js"></script>

Justification :

- Adaptation à la nouvelle architecture modulaire
- Respect des dépendances entre modules