|  |  |  |
| --- | --- | --- |
|  |  | Manuel programmeur (serveur) |

Version 1.0

Objectif 2

PrÉ-requis 2

avertissement 2

I/ première étape : qg.php 3

1) Synchronisation des joueurs 3

2) Les paramètres joueurs 3

3) Création du joueur 3

II/ Deuxieme etape : start.php 4

1) Arbitrage 4

2) Synchronisation des joueurs 4

III/ Jouez ! 5

1) Loc.php 5

2) Regle.php 5

# Objectif

1. Comprendre le déroulement d’une partie du point de vue serveur.
2. Ajuster les paramètres selon le jeu désiré.

*N.B. : Nous ne traiterons pas des codes servant à la connexion, à la création ou à la destruction d’un compte. Vous pouvez vous en inspirer pour créer le votre, utiliser vos propres codes ou vous référer au manuel d’utilisateur (serveur) V1.0 pour une simple migration vers vos serveurs.*

# PrÉ-requis

* Des connaissances en PDO sont fortement recommandées.  
  *Documentation et aide disponible sur :*
  + *http://php.net/manual/fr/book.pdo.php*
  + *http://php.developpez.com/faq/?page=pdo*
* Des connaissances dans les composantes géographiques sur serveur sont utiles.

# avertissement

* Ces codes peuvent (et devront) être améliorés. Cependant, la trame du jeu est claire et cohérente.
* Les améliorations sont :
  + Mettre les règles cohérentes sur toute la surface de la Terre.
  + Revoir la gestion de la protection (trop puissant ici).
  + Améliorer l’enchainement des parties si plusieurs veulent être lancées au même instant.
* Les codes sont déjà commentés. Je ne reviendrai pas en détail sur ceux-ci. Ce manuel à pour but de comprendre les fichiers dans leur globalité et non pas ligne après ligne.
  + Les paramètres de base sont en orange.
  + Les paramètres à changer sont en vert.
* Attention ! Toute modification annexe non présentée dans ce manuel peut entrainer des erreurs.

# I/ première étape : qg.php

## Synchronisation des joueurs

Lors de votre connexion, vous devez attendre vos futurs alliés/ennemis. Il vous faut donc patienter. Cette attente se déroule dans le premier tiers du fichier. Ici, le nombre de joueur dans la partie sera de 6.

Vous pouvez choisir ici le nombre de joueurs pour votre partie. Pour ce faire, il faut modifier la limite de la variable $rows['d\_sync'].  
Exemple : $rows['d\_sync']<10 si vous voulez jouer à 10.

N.B. : Vous pouvez également changer le temps d’endormissement entre chaque requête vers le serveur grâce à la fonction sleep(). Le nombre indique des secondes.

## Les paramètres joueurs

Lorsqu’un joueur arrive et se connecte, son identifiant correspond à l’enchainement des connexions. Le premier connecté est le joueur 1, le deuxième le joueur 2…

Attention ! Les joueurs 1 et 2 seront respectivement chef de l’équipe 1 et 2 !

Les autres joueurs sont répartis comme suit :

* Id pair 🡺 équipe 2
* Id impair 🡺 équipe 1

Le joueur possède donc 4 paramètres fondamentaux : son id dans la partie, l’id de la partie (attribué automatiquement lors de la connexion), s’il est chef ou non et son équipe.

## Création du joueur

Chaque joueur possède donc ces 4 propres paramètres. Ils sont donc envoyés vers le serveur des comptes (il pourrait être également envoyé vers le serveur de jeu). Dans le serveur de jeu, chaque joueur reçoit donc 100 points de capa et renvoie 3 paramètres fondamentaux (le chef d’équipe étant l’id 1 ou 2, on peut s’affranchir de ce paramètre). La valeur 100 peut être ajustée en fonction de votre partie.

Le compteur de joueur est remis à zéro. Une nouvelle partie peut commencer.

# II/ Deuxieme etape : start.php

## Arbitrage

L’arbitre (ici le joueur 1) positionne les bases et batteries des deux équipes. Il envoie 2 listes avec les positions et les équipes de ces objets. Les positions sont exprimées en longitude et latitude. Elles sont envoyées dans un objet POINT au serveur.

Attention ! Le POINT est à remplir POINT (longitude, latitude) ! Non l’inverse !

Chaque batterie et base se voient attribuer 1000 points de capa également ajustable en fonction de votre partie.

## Synchronisation des joueurs

Pendant le temps de la mise en place du jeu, les autres joueurs sont dans ce même fichier. Ils attendent les 6 joueurs, s’il y en a 6 sinon $rows['s\_sync'] est à ajuster de la même façon que $rows['d\_sync'] (Voir 1/PREMIERE ETAPE : QG.PHP).

Le début de la partie démarre 30 secondes après la fin de la mise en place du jeu. Il fait appelle à la fonction time(), exprimée en seconde depuis de début de l’époque UNIX. Il est identique pour tout le monde et tous les joueurs pourront débuter la partie ensemble.

Le compteur de joueur est à son tour remis à zéro. Une nouvelle mise en place de partie peut commencer.

Fin de la mise en place

# III/ Jouez !

Le jeu commence. Deux fichiers seront appelés successivement : loc.php et regle.php. La séquence est à peu près comme celle-ci et se répète :

Loc | Loc | Loc | Règle

## Loc.php

Le fichier récupère les positions des joueurs à intervalle régulier, les stockent et les renvoient au joueur.

## Regle.php

Ce fichier est appelé par tous les joueurs. Pour faire preuve d’équité, à chaque séquence, le joueur exécutera une règle. La séquence suivante démarre quand toutes les règles ont été exécutée.

Tout d’abord, la table de règle est réinitialisée.

Ensuite, les règles sont exécutées en fonction de leur ordre d’arrivée. En fait, il met à jour la table des règles. La régénération en capa est systématique.

* Observation ajoute 50m de visibilité maximum 150m.
* Assaut sélectionne le joueur ennemi le plus proche dans un rayon de 18 (degrés à priori !), blesse ce joueur de capa\_assaillant/4 et reçoit des blessures de défense de capa\_ennemi/10.
* Tir reçoit du joueur un cap de tir, sélectionne tous les joueurs et base ou batteries ennemis dans un angle de + ou – 45 degrés d’une distance inférieure à 100 (degrés à priori !) et leur inflige un dégât inversement proportionnel à la distance.
* Récupération permet de récupérer davantage de capa (10 fois plus que d’ordinaire) mais multiplie par 10 les dégâts reçus.

Tant que les 6 joueurs n’ont pas exécuté leur règle, ils attendent.

N.B. : si le nombre de joueurs est différent de 6, mettre $compte > nb de joueurs -1

Le calcul est ainsi déclenché. Il s’agit d’une somme de dégâts et de soins. Le nombre de capa d’un joueur ne peut excéder 100 ou la limite que vous vous fixez. Dans ce cas $capa > Valeur de capa fixée et $capa = Valeur de capa fixée.

On fait attendre les joueurs le temps que tous les calculs soient effectués. On sélectionne les capa des bases et batteries ainsi que le rayon d’observation et les capa du joueur. Une nouvelle séquence peut démarrer.