Le RegisterPacket

kazuma

Salut à tous, aujourd'hui je désire partager avec vous ce que je sais à propos du fonctionnement d'une fonction pour les packets de l'API Eluna, plus précisément la fonction "RegisterPacketEvent", mais tout d'abord je vous expliquerai à ma manière ce que sont les packets.

Tout d'abord je tiens à préciser que ce que je vais vous expliquer c'est tout ce que je découvert par moi-même, alors je tenterai de vous expliquer à ma façon ces fonctions, c'est donc possible que je n'utilise pas les bons termes techniques. Aussi, je n'empêche personne d'apporter sa pierre à l'édifice s'il en sait davantage.

Qu'est-ce que c'est un packet

Dans cette partie je vous ferai une brève explication de ce qu'est un packet, ce que fais un packet , à quels moments , comment etc sans pour autant trop entrer dans les détails.

Premièrement, imaginons un packet comme un message, il n'existe pas que dans WoW mais dans une multitude de systèmes informatiques. C'est un message qui est envoyé d'une machine à une autre , comme deux individus s'envoyant des lettres.

Dans le cas de WoW , les packets sont envoyés à partir de deux endroits différents, il y a le client et le serveur.
Pour être plus clair pour certains c'est le principe du : Maître - Esclave

Le maître : Le serveur , donc votre serveur WoW installé sur votre machine (Vps , serveur dédié ou voir même votre propre ordinateur si vous êtes en local).

L'esclave : Le client (donc le jeu du joueur , ce qui lui permet de jouer) présent sur la machine de ce dernier souhaitant se connecter à votre serveur WoW.

Ces deux machines communiquent en s'envoyant des messages (packet).

Qu'est-ce qu-ils se disent ?

Bonne question, enfaite c'est très simple, le client dit au serveur ce que fais le joueur ou ce qu'il demande, et le serveur réponds suivit de l'action par rapport à la situation.

Exemple concret :
Le client :

> Serveur, le joueur souhaite ouvrir son inventaire..

Le serveur :

> D'accord client, je sais ce qu'il me reste à faire présente l'inventaire aux yeux du joueur.

Le client :

> Serveur, le joueur cible quelqu'un..

Le serveur :

> D'accord client, je sais ce qu'il me reste à faire fais apparaître le portrait de la cible aux yeux du joueur.

Et ainsi de suite, énormément de packet sont envoyé par seconde, et encore plus selon le nombre de joueurs qui jouent. Donc ces deux la sont très bavard !

Tout cela grâce aux packet, et nous en tant que développeurs nous avons aussi la possibilité de manipuler tout cela facilement grâce à l'API

Eluna.

Mais comment différencier les messages ?

Blizzard a créé son propre message pour que les machines se parlent et puissent se comprendre, les packets contiennent quelque chose qui permet aux machines de savoir de quoi parle leur interlocuteur , ce quelque chose s'appel l'**Opcodes**.

Comment fonctionne l'Opcodes ?

Il y a une multitude d'opcodes et il y en a toujours de nouveaux à chaque nouvelle extensions du jeu. Les opcodes sont les différents messages que s'envoient le serveur et le client. C'est-à-dire que les machines se disent des choses préparées quoi.

Ça veut dire qu'ils ne peuvent pas dire n'importe , ils disent seulement ce qu'ont leur à appris.

Voici un site contenant une liste de beaucoup d'opcode (Je ne sais pas si il est à jour, mais il est largement satisfaisant je pense pour l'extension WoTLK)

Sur ce site on peut y voir énormément phrases différentes , il y a celles utilisables uniquement depuis le serveur, ainsi que celles utilisables uniquement depuis le client (et peut-être certaines utilisables depuis les deux côtés, je n'en ai testé que très peu...)

On y distingue des phrases commençant par C ou S comme comme ceci :

CMSG_ATTACKSTOP	0x142
SMSG_ATTACKSTART	0x143

- Celles commençant par C signifie qu'elles ne peuvent être envoyées que à partir du Client (Client)
- Celles commençant par S signifie qu'elles ne peuvent être envoyées que à partir du Serveur (Server)

D'autres phrases n'ont pas de préfixe comme "S" ou "C" , peut-être comme je l'ai dis ci-dessus qu'elles sont susceptibles d'être utilisable des deux côtés Serveur ou Client, il suffira de tester pour savoir !

Quelle en est l'utilité ?

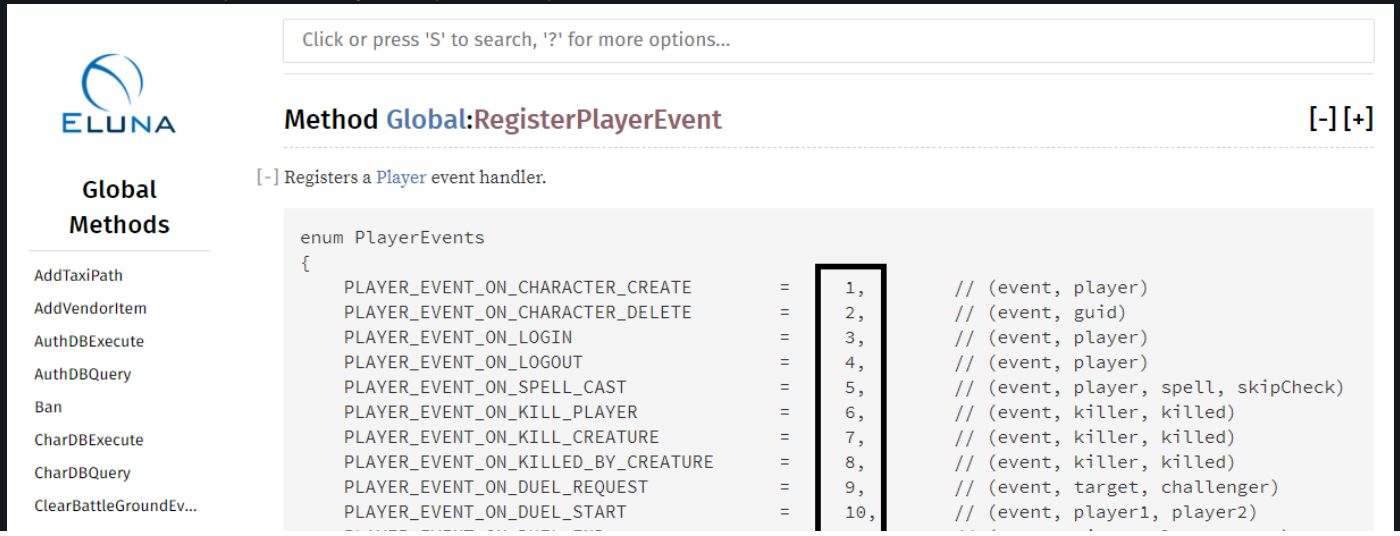
En sachant manipuler les opcodes cela pourrait vous créer de nouvelles opportunités, car au final si vous savez à quel moment précis l'action qu'à fait un ou des joueurs alors vous pouvez baser votre code Lua sur tout ce que vous voulez, **vous sentez ces nouvelles opportunités ?**

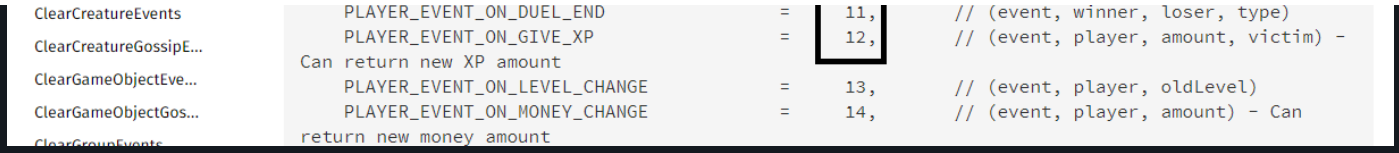
Toujours pas ? alors je vais me montrer plus clair.

L'Api Eluna contient un certain nombre de Hook, les événements détectés qui vous permettent ensuite de caler votre code , vos fonctions pour qu'elles se déclenchent au moment approprié.
Vous y arrivez grâce aux **RegisterServerEvent, RegisterPlayerEvent, RegisterGuildEvent, RegisterCreatureEvent etc...**

Enfaite si vous cherchez bien dans les fichiers de votre serveur Wow, on peut même y constater que les fichiers codés en C++ utilisent les opcodes pour détecter eux aussi les événements précis et vous fournir par le biais de l'API Eluna les fonctions Register qui détectent ces évènements que vous pouvez préciser grâce à un numéro. Par le biais des fonctions d'Eluna c'est simple car les fichiers C++ qui ont créé Eluna vous mâchent le travail et vous comprendrez pourquoi...

Voici les numéros en question dont je vous parle, vous permettant de choisir les événements :





L'utilité au final c'est que maintenant si vous ne trouvez pas l'événement précis qu'il vous faut pour déclencher votre code Lua. Vous le trouverez sûrement directement dans cette liste d'opcode. Bien que Eluna vous met à disposition beaucoup d'événement selon des cas bien spécifique peut-être que ça peut valoir le coût de chercher directement l'opcode correspondant à votre souhait.
J'espère être clair et ne pas en embrouiller certains...

Comment s'en servir ?

Bien heureusement, ceux qui ont créé Eluna ont pensé à nous mettre à disposition une façon d'utiliser directement les opcodes contenus dans les packets et manipuler les packets, c'est génial , non ?

On commence à arriver au niveau du code maintenant !
Vous en savez maintenant suffisament je pense pour utiliser le RegisterPacketEvent. Enfaite comme son nom l'indique , on va pouvoir assigné du code lua directement à un packet.

Donc après avoir créer votre fonction en lua , vous allez l'assigner à un événement donc voici la documentation pour la syntaxe:


```
RegisterPacketEvent( entry, event, function )
```

- **Entry** : *On peut voir que en entry, la fonction attend un opcode , donc c'est ici qu'on mettra l'opcode donc il suffit de mettre le numéro qu'on voit à la droite de chaque phrase sur le site que je vous ai envoyé , MAIS ce n'est pas fini ! Il faut quand même un semblant de difficulté sinon c'est trop facile...*

Exemple : 0x143 , l'opcode qui correspond à "Serveur : L'attaque à commencé !" est actuellement en base 16 , la numérotation hexadécimal , vous devez d'abord la convertir en base 10 c'est à dire décimal , les nombres que nous utilisons normalement tout les joueurs.

Vous avez le choix entre le convertir à la main (voir de tête) si vous êtes un super informaticien sinon il existe des sites pour encore vous mâcher le travail et convertir à votre place.

Pour ma part j'utilise ce site : **Wims.unice**

Il vous suffira d'entrer tout la partie après le "0x" donc 143 par exemple si on prend pour exemple : 0x143 , d'indiquer que c'est écrit en base 16 (héxadécimale) pour le convertir en base 10 (décimale) , et vous obtiendrez un tout autre nombre, celui la c'est la valeur que vous devez renseigner pour entry.

La fonction veut l'opcode que en base 10.

- **Event** : *le code provient du serveur alors on se positionne comme si on était le serveur*
  On a le choix entre **PACKET_EVENT_ON_PACKET_RECEIVE** et **PACKET_EVENT_ON_PACKET_SEND**.
  Donc si c'est un message qui provient du client on va choisir 5 pour **PACKET_EVENT_ON_PACKET_RECEIVE** sinon si c'est le serveur qu'il l'envoie c'est l'autre 7 pour **PACKET_EVENT_ON_PACKET_SEND**.
- **function** : *c'est simplement le nom de votre fonction, vous le savez déjà.*
- **Shots** : *le nombre de fois que ce déclenche la fonction, ça vous le savez déjà et c'est facultatif si c'est infini.*

Voici un exemple d'utilisation :

```
local function hautfait(event, packet, player)
    SendWorldMessage(player:GetName().." à réussi un haut fait !")
end

RegisterPacketEvent(1128 ,7, hautfait)
```

Une fonction qui indique à tout le monde que tel joueur à accompli un haut-fait.

J'ai utilisé cet opcode :
SMSG_ACHIEVEMENT_EARNED 0x468
Comme c'est un message utilisable que par le Serveur, j'ai eu recours au hook **PACKET_EVENT_ON_PACKET_SEND (7)** et j'ai bien sûr converti **468 (base 16) en base 10 (1128).**

Super maintenant, vous avez désormais toutes les cartes en mains pour pouvoir vous servir de cette fonction et peut-être même commencer à utiliser les autres !
